Valentin Ivașcu
"oriceon"

v 2.1



INIȚIERE ÎN PHP & MySQL

036 05684169 A 123 H56 00001003
publicat 01.06.2005
ultima actualizare 28.10.2005

PENTRU PHP 4

V 2.1

CUPRINS

Cateva cuvinte din partea autorului

In primul rand as vrea sa va multumesc pentru faptul ca sunteti interesati de aceasta lucrare.

In acest tutorial va sunt prezentate bazele limbajului de programare PHP, precum si ale bazei de date MySQL. Tutorialul este destinat persoanelor care au deja cunostinte de **HTML** (**H**yper**T**ext **M**arkup **L**anguage), si vor sa realizeze pagini web dinamice care sa interactioneze cu utilizatorul, **PHP** fiind un limbaj destul de puternic, cu ajutorul caruia puteti dezvolta aplicatii web foarte complexe si interesante.

O data invatat acest limbaj, puteti spune adio modificarilor HTML pentru a introduce noi texte, poze, informatii in pagina dumneavoastra.

Puteti sa va creati un sistem de administrare dimanic, cu ajutorul caruia sa va modificati paginile online. Ceea ce vreau sa va mai spun este faptul ca toate exemplele ce vor urma sunt scrise din prisma stilului meu de a coda, nu luati ca pe un standard, ci doar ca pe un stil al meu.

Pentru o mai buna realizare a acestui tutorial, va rog frumos sa imi trimiteti un e-mail la adresa oriceon@yahoo.com cu ceea ce ati dori sa mai apara in tutorial, cu ce exemple, cu ce informatii, ceea ce ati vrea sa stiti/faceti si nu puteti.

Urmariti cat mai des adresa tutorialului http://www.oriceon.com/tutorial/ pentru a descarca noile versiuni. Cand descarcati o noua versiune, cititi-o si faceti o comparatie cu cea veche, deoarece este posibil sa se fi strecurat mici greseli in versiunea veche, asta pe de o parte, iar pe de alta parte, apar noi lucruri in noile versiuni, nu numai exemple.

Ca un sfat pe care as putea sa vi-l dau, pentru inceput, ar fi acela sa invatati sa scrieti codurile cat mai frumos, fara prea multe spatii, din cat mai putine linii.

In speranta ca acest tutorial vå va fi de folos, si cå va veti pune si dumneavoastra mintea la contributie, va doresc invatare placuta si astept cu nerabdare mesaje de la voi.

Discutii legate de acest tutorial mai puteti sa le faceti si pe adresele:

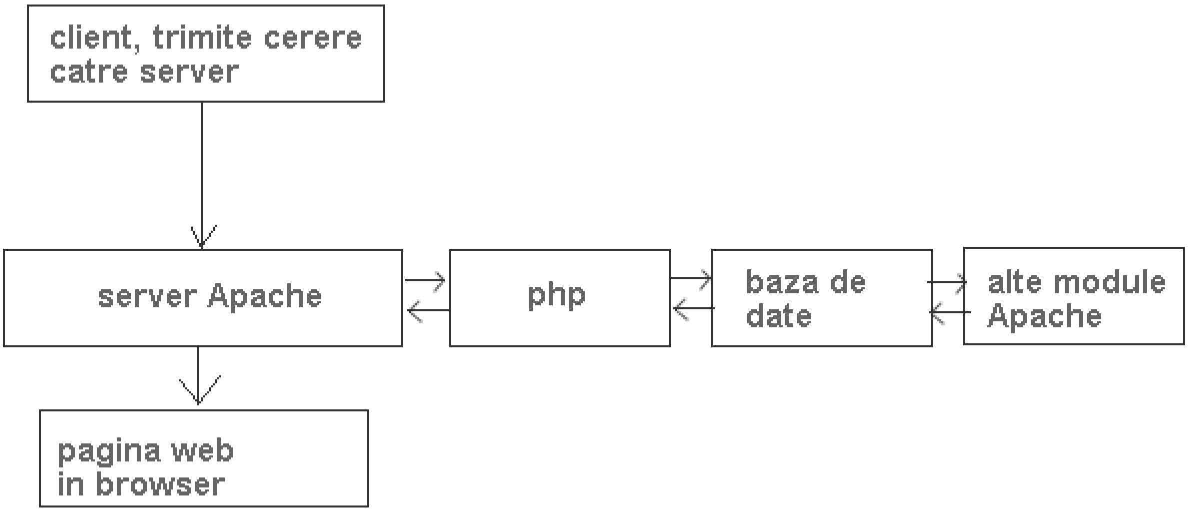
http://forum.phpromania.net/viewtopic.php?t=2684
http://forum.softpedia.com/index.php?showtopic=61508

Ivascu Valentin (oriceon)

Ce este PHP ?

PHP (se pronunta pe-haş-pe) este un limbaj de programare ce ruleaza server, proiectat special pentru WEB. Intr-o pagina HTML puteti ingloba cod PHP care va fi executat la fiecare vizitare a paginii.

Codul dumneavoastra PHP este interpretat pe serverul WEB si genereaza un cod HTML care va fi vazut de Uilizator (clientului (browserului) fiindu-i transmis numai cod interpretat ca si HTML).



PHP a fost conceput in anul 1994 si a fost initial munca unui singur om, **Rasmus Lerdorf**.
A fost adoptat de alti oameni talentati si a trecut prin trei rescrieri importante pentru a ajunge la produsul clar si matur de astazi. In octombrie 2002, era in uz de mai mult de noua milioane de domenii din lumea intreaga, iar acest numar este intr-o continua crestere. Numarul actual al acestora il puteti vedea la adresa http://www.php.net/usage.php

PHP este un produs Open Source, cu acces la codul sursa. Il puteti folosi, modifica si redistribui, toate acestea in mod gratuit.

Initial, PHP era acronimul de la **P**ersonal **H**ome **P**age, dar a fost modificat pentru a se alinia la conversia de numire recursiva GNU (GNU = Gnu`s Not Unix) si acum este acronimul pentru **P**HP **H**ypertext **P**reprocessor.

Versiunea actuala a **PHP** este **5.1 (RC 1)**
Pagina de baza pentru PHP este: http://www.php.net
Pagina pentru Zend (compania a carei fondatori au proiectat PHP4) se afla la http://www.zend.com

Ce este MySQL ?

Mysql (se pronunta mai-es-chiu-el) este un sistem de gestiune a bazelor de date, foarte rapid si robust.
O baza de date va permite sa stocati, sa cautati, sa sortati si sa va regasiti datele in mod eficient.
Serverul MySQL controleaza accesul la datele dumneavoastra pentru a garanta ca mai multi utilizatori pot lucra simultan cu acestea.

Deci, MySQL este un server multi-user (mai multi utilizatori) si multi-thread (mai multe fire de executie).
Utilizeaza SQL (**S**tructured **Q**uery **L**anguage), limbajul standard de interogare a bazelor de date din intreaga lume.

MySQL este disponibil in mod public din 1996, dar istoria dezvoltarii sale incepe in 1979. A castigat de mai multe ori Linux Journal Readers` Choice Award (Premiul cititorilor).

MySQL este disponibil sub o licenta Open Source, dar daca este nevoie sunt disponibile si licente comerciale.

Versiunea actuala a **MySQL** este **5.0.11** (beta).
Pagina de baza pentru MySQL este: http://www.mysql.com

Solutia completa

1) Apache

Pentru a rula un site aveti, in primul rand, nevoie de un server **HTTP** (**H**yper**T**ext **T**ransport **P**rotocol).
Alegerea mea este indreptata spre Apache datorita flexibilitatii sale, portabilitatii, sigurantei si extensibilitatii.

Versiunea actuala a **Apache** este **2.1.6** (alfa)
Pagina de baza pentru Apache este: http://httpd.apache.org/

Descarcare Apache

Sistem Linux: http://apache.iasi.roedu.net/httpd/httpd-2.0.54.tar.gz
Sistem Windows: http://apache.iasi.roedu.net/httpd/binaries/win32/apache_2.0.54-win32-x86-no_ssl.msi

Documentatie

http://httpd.apache.org/docs/2.0/

2) PHP

Descarcare PHP

Sistem Linux: http://ro.php.net/distributions/php-4.4.0.tar.bz2
Sistem Windows: http://ro.php.net/distributions/php-4.4.0-Win32.zip

Documentatie

Romana: http://ro.php.net/distributions/manual/php_manual_ro.chm
Engleza: http://ro.php.net/distributions/manual/php_manual_chm.zip

Unelte si scripturi folositoare

In primul rand va recomand scriptul **phpMyAdmin** acest script va ajuta sa va conectati la baza de date, sa o gestionati, toate acestea putand fi executate direct din pagina web.

Descarcare http://kent.dl.sourceforge.net/sourceforge/phpmyadmin/phpMyAdmin-2.6.4-pl3.zip

Pagini folositoare cu documentatii in limba engleza si scripturi PHP:
http://www.hotscripts.com/PHP/index.html
http://www.phpfreaks.com/
http://php.resourceindex.com/
http://www.onlyphp.com/
http://www.scripts.com/php-scripts/

Un SOC (**Si**stem de **O**rganizare a **C**ontinutului = **C**ontent **M**anagement **S**ystem) - realizat de un roman, este usor, frumos si cuprinde foarte multe facilitati.
Il puteti descarca de la adresa http://cms.punctweb.com/

Cea mai mare comunitate a programatorilor PHP din Romania http://www.phpromania.net unde puteti gasi informatii PHP si un mare forum (lista de discutii).

3) MySQL

Descarcare MySQL

http://dev.mysql.com/downloads/mysql/4.1.html

Documentatie

ftp://ftp.roedu.net/pub/mirrors/ftp.mysql.com/Downloads/Manual/manual.zip

4) CuteFTP

Pentru a accesa fisierele aflate pe site, trebuie sa va conectati la FTP (**F**ile **T**ransfer **P**rotocol).
Un client FTP se foloseste in cazul in care site-ul nu este gazduit pe calculatorul dumneavoastra (local).

Va recomand CuteFTP dezvoltat de GlobalScape
Versiunea actuala a **CuteFTP** este **7.1**

Descarcare CuteFTP

ftp://ftp.globalscape.com/pub/cuteftppro/cuteftppro.exe

Documentatie

http://help.globalscape.com/help/guides/CuteFTP_Pro7_User_Guide.pdf

Recomandare:

Daca rulati Windows 98/ME/XP ca sistem de operare si doriti sa va instalati un pachet intreg Apache+PHP+MySQL+phpMyAdmin pe calculatorul dumneavoastra, va recomand **EasyPHP 1.8**.

Il puteti descarca de la adresa http://ovh.dl.sourceforge.net/sourceforge/quickeasyphp/easyphp1-8_setup.exe

Scripturile PHP se scriu in orice editor de text insa pentru o mai buna vizualizare a codului php scris, aveti nevoie de un editor de text profesional.

Va recomand sa folositi **PHPEdit 1.2** pe care il puteti descarca de la adresa
http://www.waterproof.fr/products/PHPEdit/download.php

Instalare

Recomandare

Daca nu vreti sa aveti atata bataie de cap, si doriti sa instalati mai repede un pachet complet ce sa contina Apache + PHP + MySQL + phpMyAdmin, va recomand pachetul **EasyPHP**

Acest pachet nu necesita decat o instalare simpla (ca a oricarui program) si il puteti descarca de la adresa: http://ovh.dl.sourceforge.net/sourceforge/quickeasyphp/easyphp1-8_setup.exe



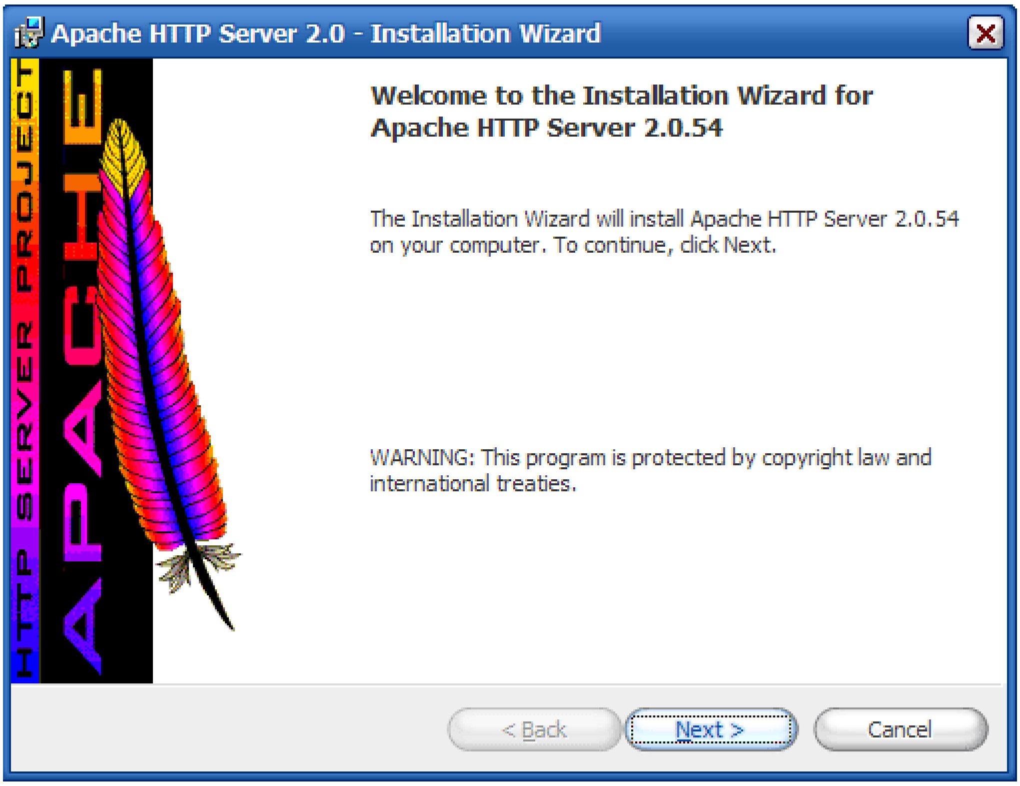
Eu folosesc acest pachet si sunt foarte multumit de el, de aceea vi-l recomand cu cea mai mare placere.

Retineti faptul ca in acest tutorial o sa lucram cu pachetul instalat in C:\Program Files\EasyPHP-1.8\ iar directorul unde sunt tinute si de unde se acceseaza paginile si scripturile PHP este www.

Daca totusi doriti sa le instalati manual, urmati pasi de mai jos.

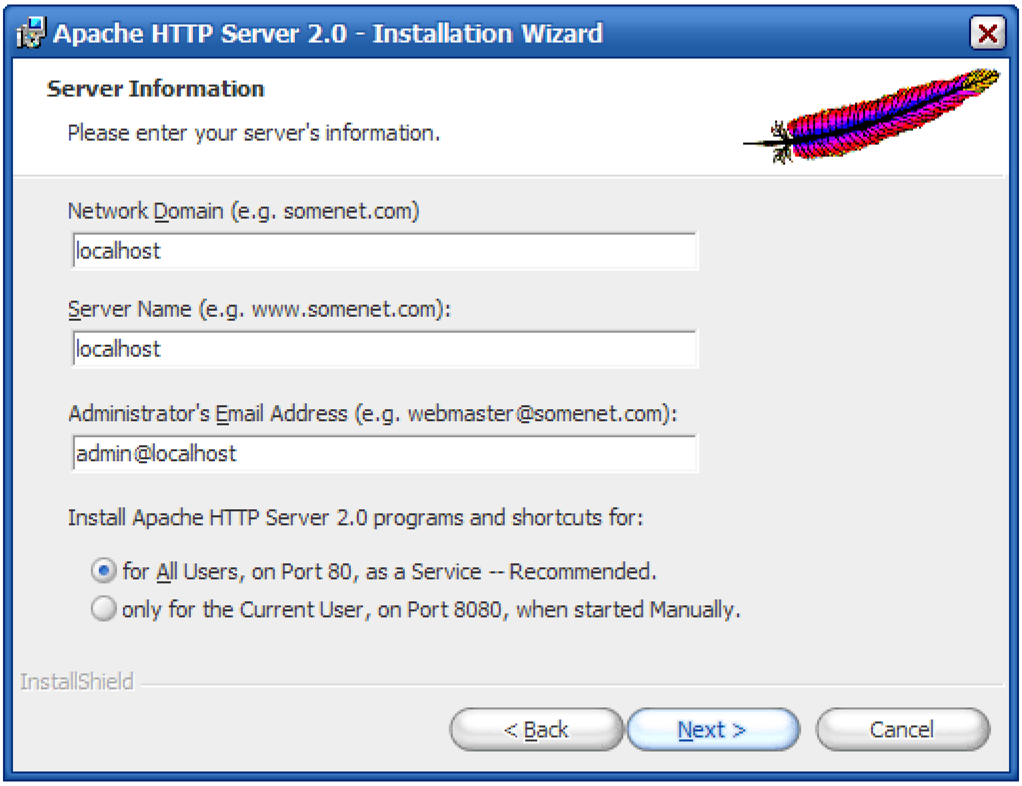
Instalare Apache

Dupa ce ati descarcat Apache de la http://apache.idilis.ro/httpd/binaries/win32/apache_2.0.54-win32-x86-no_ssl.msi, executati pentru a porni instalarea.



Apasati next, apoi (dupa ce ati citit textul) selectati optiunea “I accept the terms in the license agreement” – aceasta optiune inseamna ca ati citit si acceptat termenii din licenta si ca puteti trece mai departe la instalare apoi apasati next.

Apasati din nou next si va apare o fereastra in care completati campurile, exemplu:



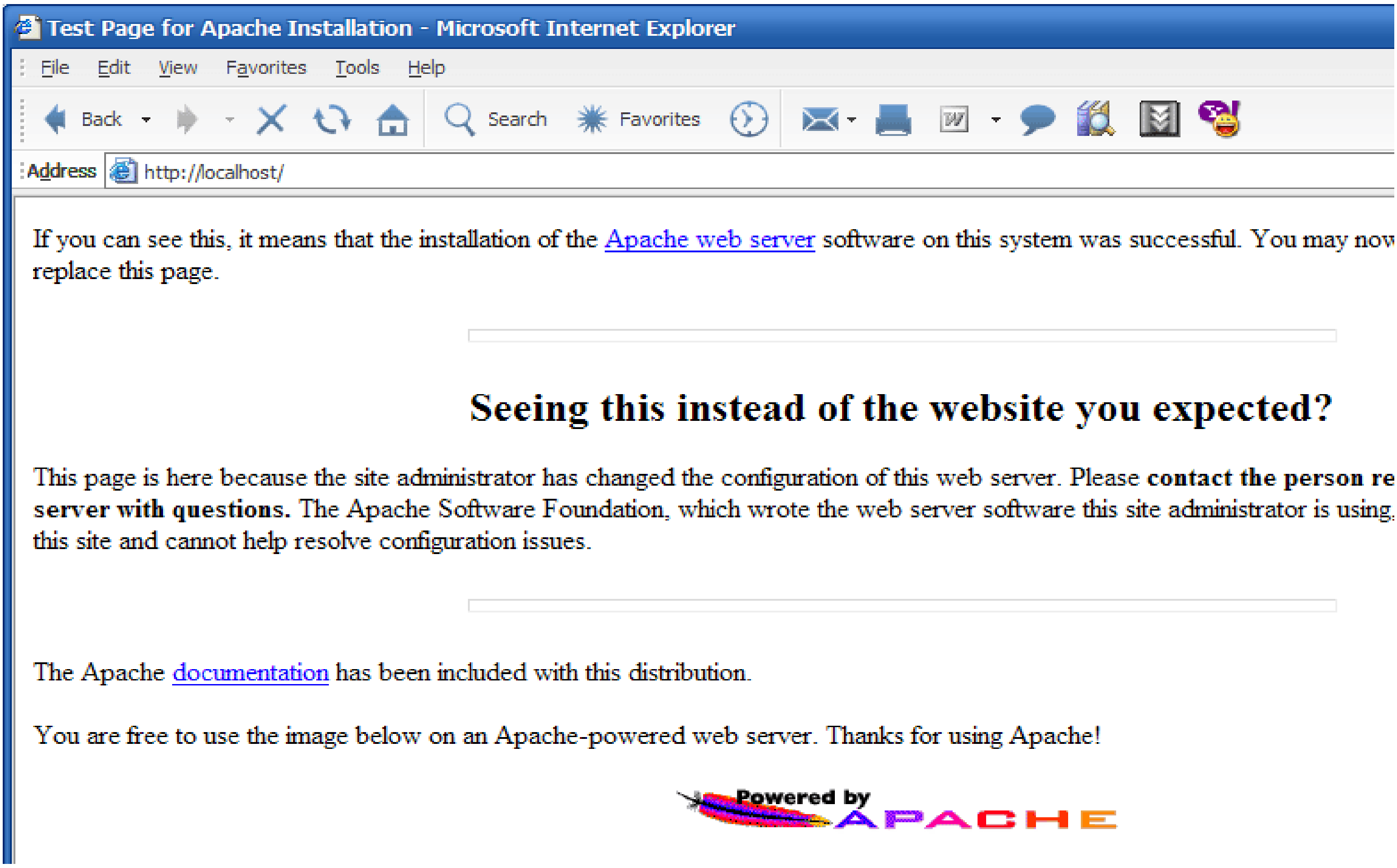
Apasati next, next, next, install si asteptati sa se efectueze instalarea apoi apasati finish pentru a termina.

Daca serverul s-a instalat cu succes, uitati-va in bara de jos (dreapta), langa ceas. Observati ca a aparut o iconitza cu un cerc, in mijlocul careia apare o sageata de culoare verde (in caz ca serverul ruleaza) sau de culoare rosie (in caz ca serverul nu a putut sa porneasca).





Accesati http://localhost/ Daca pagina arata ca cea de mai jos, inseamna ca serverul Apache a pornit si functioneaza.

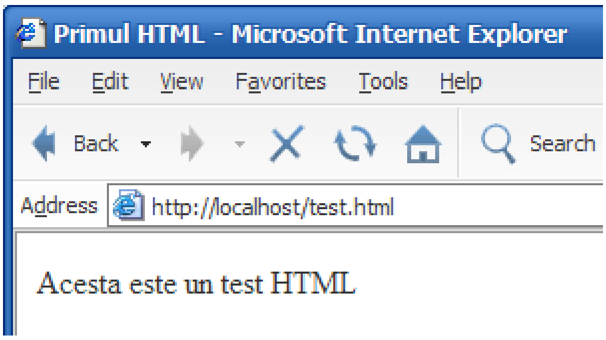


Fiecare server/pachet instalat are cate un fisier de configurare. Fisierul de configurare al serverului Apache este **httpd.conf** pe care il puteti gasi in folderul din C:\Program Files\Apache Group\Apache2\conf\

Intrati in C:\Program Files\Apache Group\Apache2\htdocs\, stergeti tot de acolo si creati un fisier test.html in care introduceti codul:

```
<html>
<head>
<title>Primul HTML</title>
</head>
<body>
Acesta este un test HTML
</body>
</html>
```

Salvati si accesati in browser: http://localhost/test.html



Observati ca serverul Apache ruleaza HTML-ul.
Pentru a rula scripturi PHP trebuie sa instalati pachetul PHP pe care l-ati descarcat.

Instalare PHP

Intrati un C:\Program Files\Apache Group\ si creati un folder cu numele php apoi dezarhivati si copiati continutul directorului php-4.4.0-Win32.

In continuare, trebuie sa editam fisierul de configurare al PHP-ului, si anume **php.ini**
Intrati in C:\Program Files\Apache Group\php, copiati fisierul php.ini-recommended si redenumiti-l in php.ini, deschideti-l in notepad si efectuati modificarile:

1) cautati **doc_root =** si schimbati cu **doc_root = "C:\Program Files\Apache Group\Apache2\htdocs"**
(aceasta operatiune se face pentru a seta calea catre folderul unde sunt tinute scripturile, paginile)

2) cautati **extension_dir =** si schimbati cu
extension_dir = "C:\Program Files\Apache Group\php\extensions"
(aceasta operatiune se face pentru a seta calea catre folderul unde sunt tinute extensiile php)

3) cautati **display_errors = Off** si schimbati cu **display_errors = On**
(aceasta operatiune se face pentru a seta ON afisarea erorilor)

4) cautati **;session.save_path = /tmp** si schimbati cu
session.save_path = "C:\Program Files\Apache Group\php\sesiuni"
(intrati apoi in folderul cu php (C:\Program Files\Apache Group\php) si creati un folder cu numele "sesiuni", aceasta operatiune se face pentru a seta calea catre sesiunile temporare de pe server)

Dupa ce ati efectuat aceste schimbari in fisierul php.ini, deschideti fisierul de config al serverului Apache, aflat in C:\Program Files\Apache Group\Apache2\conf sub numele de **httpd.conf** si efectuati urmatoarele modificari:

1) Adaugati la sfarsitul fisierului, urmatoarele 3 linii.

ScriptAlias /php/ "C:/Program Files/Apache Group/php/"
AddType application/x-httpd-php .php
Action application/x-httpd-php "/php/php.exe"

2)

Cautati linia **DirectoryIndex index.html** si schimbati-o cu **DirectoryIndex index.html index.php**

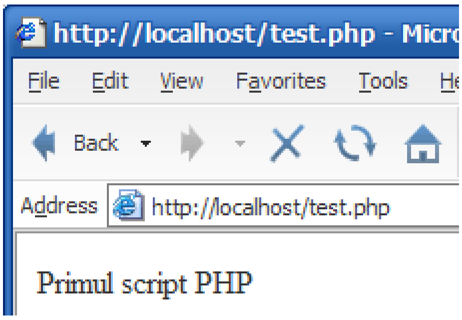
Salvati si reporniti serverul Apache.

Intrati in C:\Program Files\Apache Group\Apache2\htdocs si creati un fisier cu numele **test.php** in care introduceti codul de mai jos, salvati si accesati-l in browser: http://localhost/test.php

```
<?php

echo 'Primul script PHP';

?>
```



Daca rezultatul afisat in browser este la fel ca cel de mai sus, atunci serverul Apache + PHP s-a instalat cu succes si ruleaza.

Pentru a putea vedea configuratia php, realizati un fisier cu numele phpinfo.php, in care puneti codul:

```
<?php

phpinfo();

?>
```

Salvati si accesati in browser: http://localhost/phpinfo.php

PHP Version 4.4.0

System	Windows NT ORICEON 5.1 build 2600
Build Date	Jul 11 2005 16:08:47
Server API	CGI/FastCGI
Virtual Directory Support	enabled
Configuration File (php.ini) Path	C:\Program Files\Apache Group\php\php.ini
PHP API	20020918
PHP Extension	20020429
Zend Extension	20050606
Debug Build	no
Zend Memory Manager	enabled
Thread Safety	enabled
Registered PHP Streams	php, http, ftp, compress.zlib

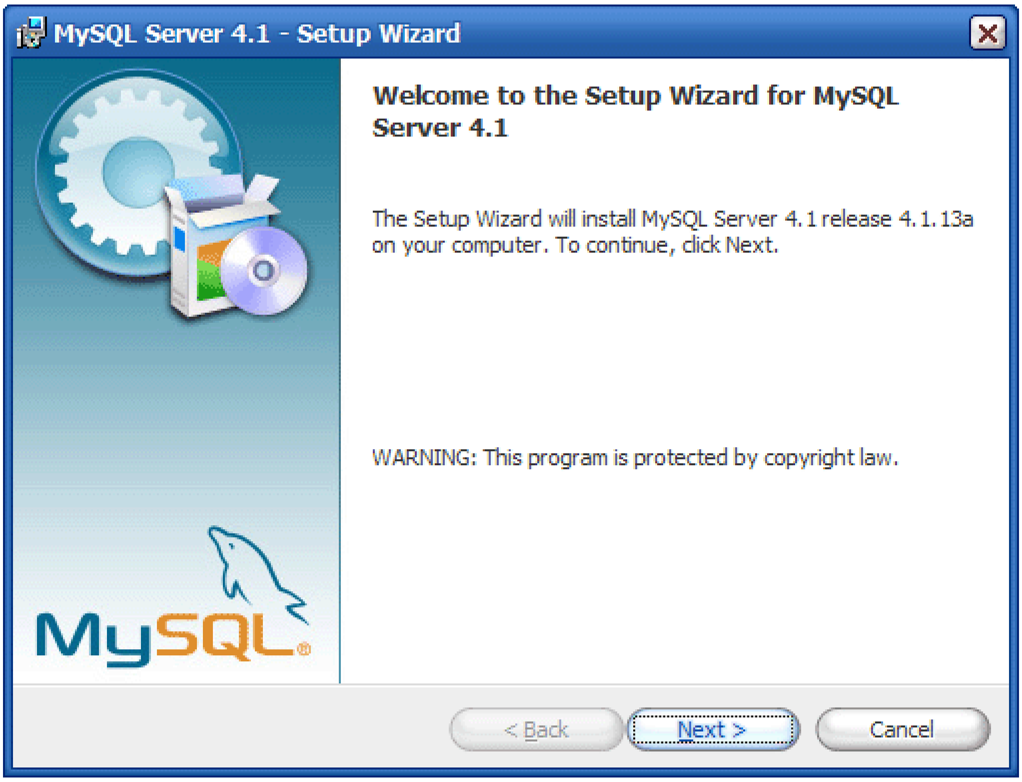
Powered by Zend This program makes use of the Zend Scripting Language Engine:
Zend Engine v1.3.0, Copyright (c) 1998-2004 Zend Technologies

In continuare, activati extensiile de care aveti nevoie stergand **;** din fata extensiei si verificand daca extensia pe care ati activat-o se afla in directorul C:\Program Files\Apache Group\php\extensions

;extension=php_gd2.dll .. stergeti ; deci:
extension=php_gd2.dll

Instalare MySQL

Dupa ce ati descarcat mysql-essential-4.1.13a-win32.msi, rulati-l, apasati next, selectati optiunea **Custom** apoi apasati next, apasati pe **Change** si alegeti calea catre C:\Program Files\Apache Group\ unde creati un folder cu numele MySQL, apasati next si install pentru a porni instalarea.



In timpul instalarii va apare o fereasta unde vå va cere sa creati un cont sau sa va autentificati la MySQL.com, selectati Skip Sing-Up si apasati butonul next apoi finish.

Dupa instalare, va va deschide o fereasta pentru a va da posibilitatea sa va configurati serverul mysql. Apasati next, apoi selectati optiunea Standard Configuration, apasati next, apoi iar next, setati o parola, cu care va veti autentifica la serverul mysql, apasati next, execute si apoi finish pentru a incheia procedeul de configurare.

Instalare phpMyAdmin

Dupa ce ati descarcat scriptul phpMyAdmin, dezarhivati si copiati folderul in directorul de unde vor rula scripturile si anume: htdocs din C:\Program Files\Apache Group\Apache2 (redenumiti folderul in phpmyadmin).

Accesati apoi http://localhost/phpmyadmin
Veti observa ca primiti o eroare..
`#1045 - Access denied for user 'root'@'localhost' (using password: NO)`

Asta inseamna ca trebuie sa editati fisierul de configurare al scriptului phpMyAdmin si sa ii setati parola (pe care ati ales-o la inregistrarea MySQL-ului).

Intrati in C:\Program Files\Apache Group\Apache2\htdocs\phpmyadmin si deschideti fisierul config.inc.php, apoi cautati linia (`$cfg['Servers'][$i]['password']        = '';`) si scrieti parola dvs, ex:

```
$cfg['Servers'][$i]['password']       = 'parolamysql';
```

Salvati si apoi dati un refresh la pagina http://localhost/phpmyadmin

Daca rezultatul va fi:

Welcome to phpMyAdmin 2.6.4-rc1

phpMyAdmin tried to connect to the MySQL server, and the server rejected the connection. You should check the host, username and password in config.inc.php and make sure that they correspond to the information given by the administrator of the MySQL server.

Error

MySQL said:

```
#1251 - Client does not support authentication protocol requested by
server; consider upgrading MySQL client
```

inseamna ca trebuie sa updatam parola mysql.

Apasati pe butonul **start** de la windows (jos, stanga) apoi in casuta **run** scrieti **cmd**, si apasati butonul ok. Tastati comanda **cd ../../Program Files/Apache Group/MySQL/bin** apoi **mysql -u root -p** si veti observa ca vi se cere parola de mysql. Introduceti parola pe care ati setat-o in momentul instalarii serverului MySQL apoi apasati enter.

Dupa ce va autentificati, observati:

C:\Program Files\Apache Group\MySQL\bin>mysql -u root -p
*Enter password: *********
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 3 to server version: 4.1.13a-nt

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql>

Tastati comanda: **SET PASSWORD FOR root@localhost=OLD_PASSWORD('parolaDvs');** apoi apasati enter, dupa care tastati comanda **FLUSH PRIVILEGES;** si apasati din nou enter.

Inchideti fereastra apoi dati refresh la pagina http://localhost/phpmyadmin/

Scriptul se incarca insa jos de tot veti observa niste erori:

The $cfg['PmaAbsoluteUri'] directive MUST be set in your configuration file!
The mbstring PHP extension was not found and you seem to be using a multibyte charset. Without the mbstring extension phpMyAdmin is unable to split strings correctly and it may result in unexpected results.

Cautati linia $cfg['PmaAbsoluteUri'] in config.inc.php si modificati-o din `$cfg['PmaAbsoluteUri'] = '';` in `$cfg['PmaAbsoluteUri'] = 'http://localhost/phpmyadmin';`

Dati un refresh la pagina si observati ca a ramas decat eroarea legata de extensia mbstring deoarece acea extensie nu a fost activata din php.ini

Intrati in C:\Program Files\Apache Group\php si deschideti fisierul php.ini.
Cautati linia **;extension=php_mbstring.dll** si activati-o eliminand **;** din fata ei astfel incat rezultatul sa fie **extension=php_mbstring.dll**

Salvati, apoi dati refresh din nou la pagina.

Observati ca erorile au disparut, iar scriptul phpMyAdmin ruleaza fara probleme.

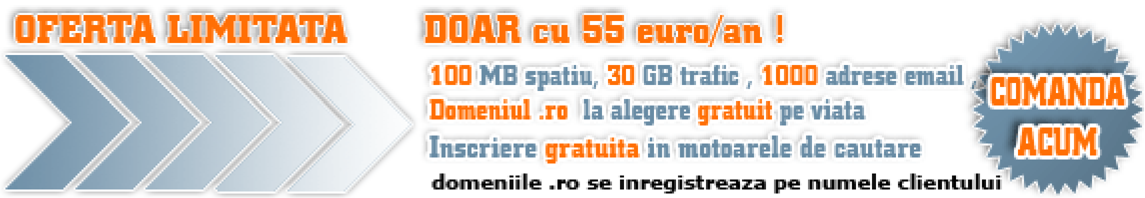
De stiut! – Trebuie sa stiti ca, pentru a da posibilitatea altor persoane sa va acceseze scripturile PHP, trebuie sa aveti un domeniu gazduit pe un server ce sa suporte PHP si MySQL.

Pe internet sunt foarte multe site-uri ce isi ofera in mod gratuit aceste servicii, insa veti avea numele gen: http://nume.3x.ro sau http://nume.as.ro/ … sau http://site.ro/nume_user .... insa toate acestea va limiteaza acces-ul la functiile PHP si multe multe altele, sau vi se introduc reclame in pagina, stricand astfel orice stil al paginii… orice frumusete…

De aceea, este bine sa va cumparati un domeniu si o gazduire buna, si aceasta la un pret foarte accesibil.

Va recomand serviciile de la http://www.mxhost.ro/ - fiind foarte accesibile oricarei persoane. Practic, doar cu un euro pe luna, aveti un domeniu al dumneavoastra **.com .org** sau **.net** vi se ofera posibilitarea crearii a 100 adrese de email @ domeniul_dumneavoastra.com, 3 baze de date MySQL, scripturi deja realizate, si multe altele.

Pentru o oferta mai detaliata si pentru restul serviciilor va rog sa vizitati http://www.mxhost.ro/





MXHOST™ http://www.mxhost.ro

Initiere in PHP

Primul exemplu in crearea unui script PHP

Trebuie sa stiti ca intotdeauna codul php este delimitat cu etichetele **<?** si respectiv **?>** sau **<?php** si respectiv **?>**

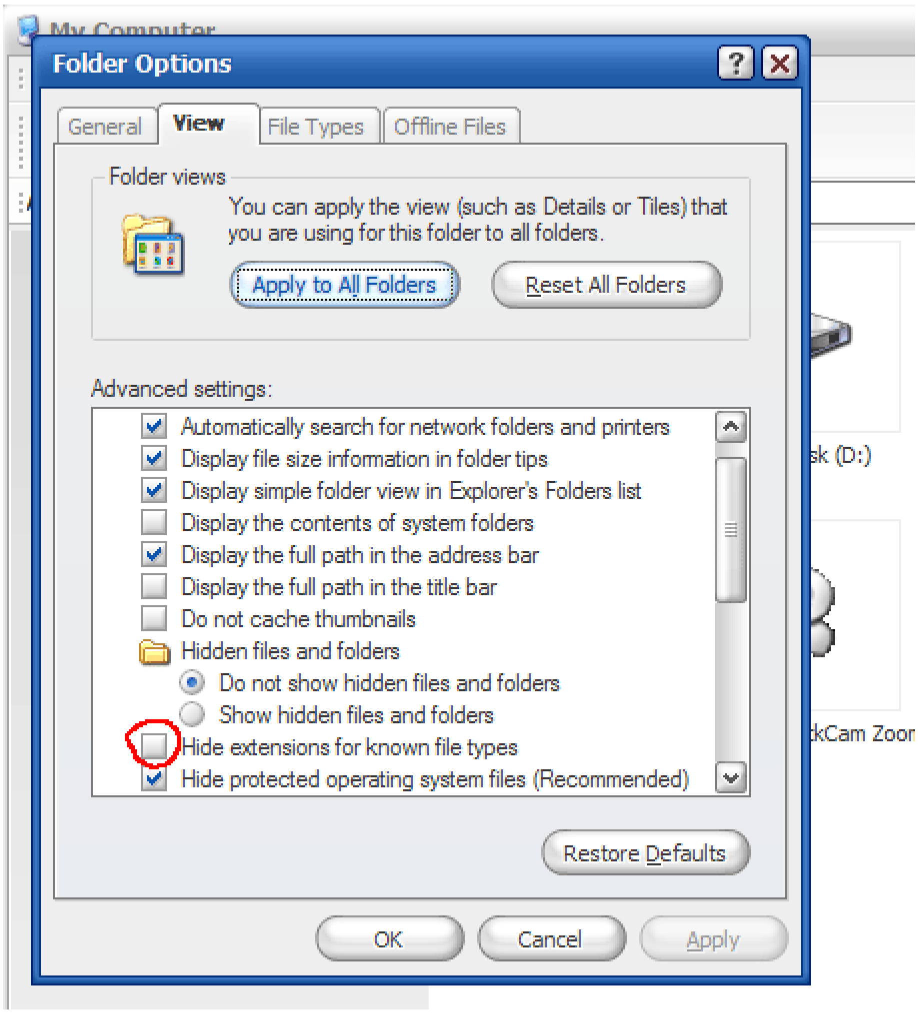
Sa luam ca exemplu urmatorul cod:

```
<?php
echo 'Salut, acesta este primul meu script PHP';
?>
```

Intram in directorul unde am instalat serverul web (in cazul nostru: C:\Program Files\EasyPHP-1.8\) si accesam directorul WWW. Acesta este directorul radacina de unde serverul nostru stie sa listeze paginile.

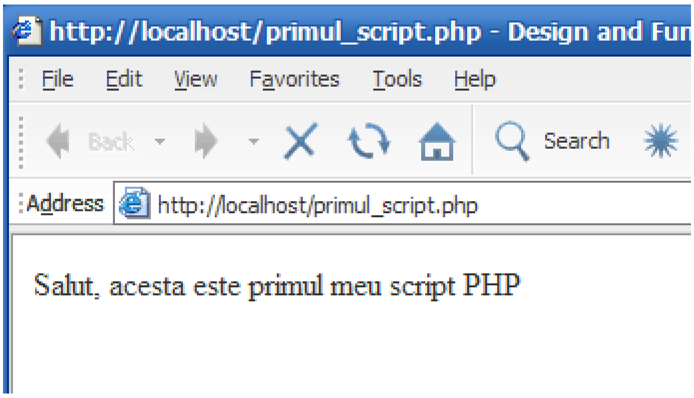
Cream o noua pagina cu numele: **primul_script.php,** insa aveti grija ca nu cumva sa aveti extensiile ascunse, deoarece, daca dati click dreapta / new / text document si il redenumiti in **primul_script.php** , acesta va avea numele de **primul_script.php.txt** deoarece extensia **.txt** este ascunsa.

Ca sa vedeti daca aveti extensiile ascunse, creati un text document apasand click dreapta / new / text document si, daca numele documentului text o sa fie New Text Document.txt, inseamna ca totul este ok, daca numele o sa fie New Text Document, inseamna ca extensia este ascunsa.
Pentru a o arata, trebuie sa facem o setare in windows si anume: deschideti **My Computer** apoi apasati sus de tot pe meniul **Tools**, apoi **Folder Options**, apoi **View**, apoi cautati unde scrie “**Hide extensions for known file types”** si debifati casuta.



Dupa aceasta mica pauza cu configurarea extensiilor in Windows, ne vom intoarce la primul nostru script PHP.

Ati creat fisierul **primul_script.php** in C:\Program Files\EasyPHP-1.8\www si pentru a-l accesa deschideti un browser (Internet Explorer sau Mozilla Firefox) si tastati adresa: http://localhost/primul_script.php

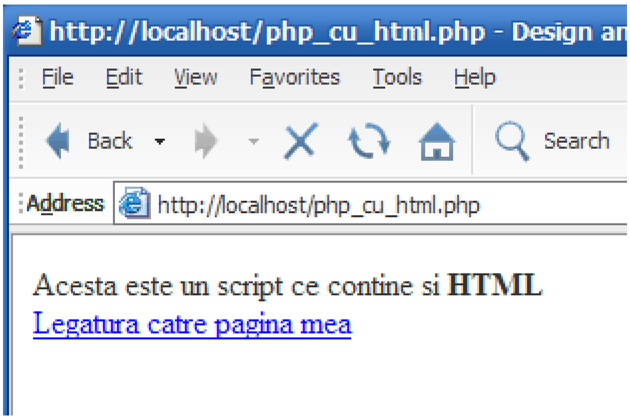


Dupa cum vedeti, textul este afisat in pagina cu ajutorul constructiei **echo,** care este delimitata de ghilimelele magice ‘. Acestea pot fi si duble “ insa in nici un caz una simpla si cea de inchidere dubla: ‘ si “ sau “ si ‘.

Este bine de stiut ca in scriptul PHP puteti ingloba si HTML si anume:

```
<?php
echo 'Acesta este un script ce contine si <b>HTML</b> <br>
      <a href="pagina_mea.php">Legatura catre pagina mea</a>';
?>
```

Realizati un fisier cu numele **php_cu_html.php** puneti codul inauntru, salvati si accesati in browser http://localhost/php_cu_html.php

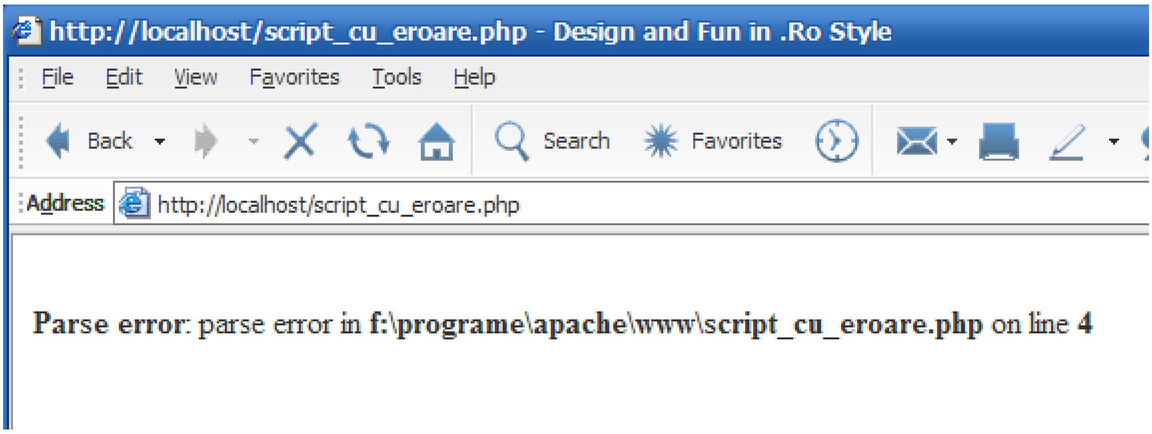


Trebuie sa stiti ca atunci cand gresiti vreo sintaxa, PHP-ul va arata o eroare care va indica unde ati gresit (pe parcurs veti invata si cum sa depanati aceste erori). Aceasta eroare apare numai in cazul in care in **php.ini** (fisierul de configurare a php-ului) are setat **display_errors = On** si **error_reporting = E_ALL**.

Revenind la erorile ce pot apare in script, in cazul in care gresiti sintaxa, luam urmatorul exemplu:

```
<?php
echo 'Acesta este un script scris gresit";
?>
```

Puneti acest cod intr-un fisier numit **script_cu_eroare.php**
Accesati scriptul: http://localhost/script_cu_eroare.php

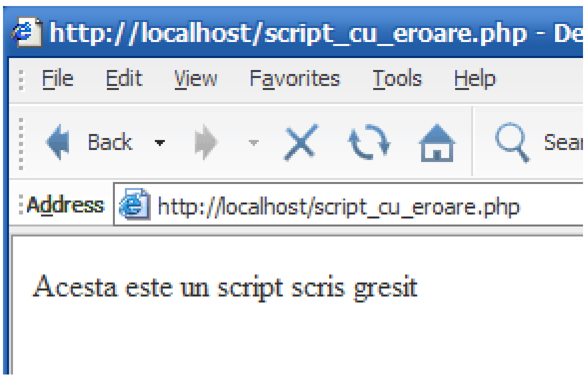


Observati de ce este bine si recomandat sa folositi un editor PHP cum este PHP Edit, deoarece linia de inchidere nu mai este rosie, ci este albastra, fapt care ne indica ca ceva nu este in regula, si ne face sa fim mai atenti la liniile din preajma ei.

Uitandu-ne la poza de mai sus observam ca primim o eroare in loc sa ni se afiseze texult “Acesta este un script scris gresit”.

Luati codul si modificati constructia echo apoi salvati si vizualizati din nou in browser

```
<?php
echo 'Acesta este un script scris gresit';
?>
```



Observati ca nu mai apare eroarea. De ce? Asta va las pe voi sa vedeti.

In constructia **echo** se poate pune orice fel de text insa aveti grija ca pot fi cazuri in care sa existe conflicte cu ghilimelele magice si anume:

```
<?php
echo 'Conflict cu ghilimelele magice <br>
      <a href='pagina_mea.php'>Legatura catre pagina mea</a>';
?>
```

Puneti codul intr-un fisier **conflict.php**, salvati si apoi vizualizati in browser accesand http://localhost/conflict.php



Dupa cum vedeti, eroarea ne spune ce este gresit in constructia noastra, si anume vedem ca exista conflict intre gilimelele magice ‘ sau “ cu ghilimele normale din interiorul constructiei echo.

Ca sa fiu mai explicit, ghilimelele din Legatura catre pagina mea fac conflict cu ghilimelele magice din delimitarea constuctiei echo: **echo ‘’;**

In acest caz, avem doua posibilitati de rezolvare a problemei si anume:

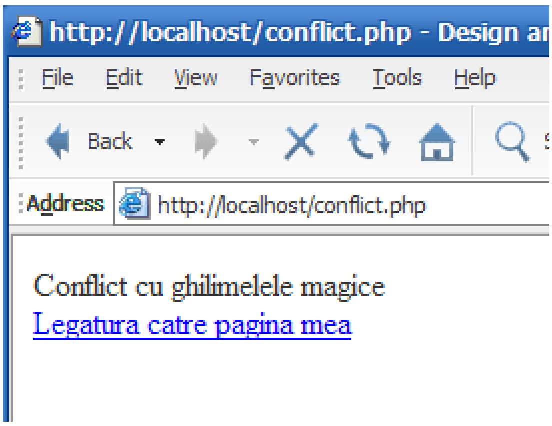
1) Putem inlocui ghilimelele din interiorul legaturii **a href** cu ghilimele duble **“ “** si asa vom scapa de conflictul cu ghilimelele magice simple din PHP.
(Ghilimelele standard din interiorul unui cod HTML sunt ghilimele duble “ insa se pot ivi cazuri in care sa aveti si ‘ insa asta mai rar).

```
<?php
echo 'Conflict cu ghilimelele magice <br>
      <a href="pagina_mea.php">Legatura catre pagina mea</a>';
?>
```

2) Putem sa le anulam folosind o linie inversa **** aplicata in fata ghilimelelor din interiorul legaturii **a href** (aceasta anuland practic acele ghilimele) si anume:

```
<?php
echo 'Conflict cu ghilimelele magice <br>
      <a href=\'pagina_mea.php\'>Legatura catre pagina mea</a>';
?>
```

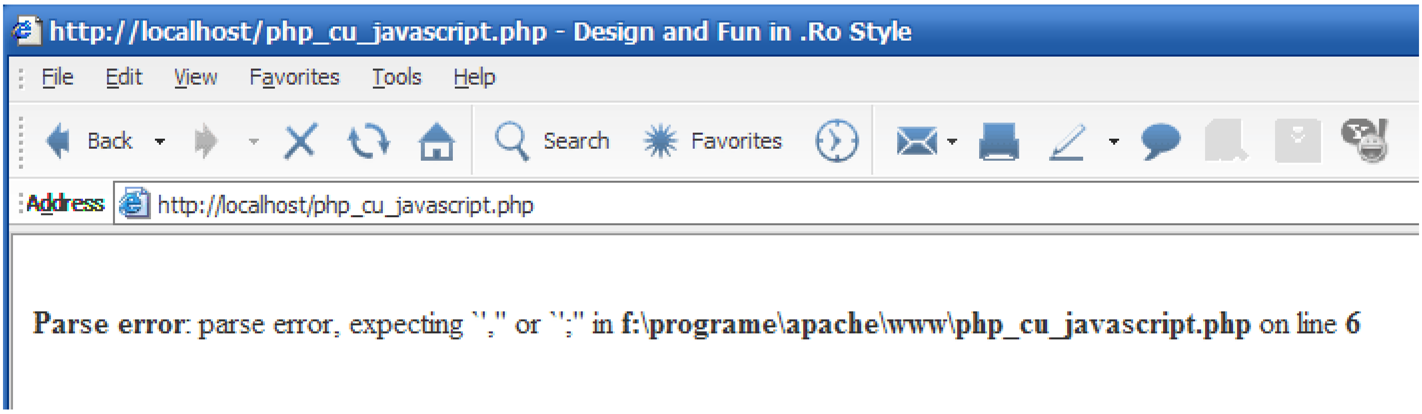
Realizati aceste modificari pe rand si vizualizati in browserul dumneavoastra.



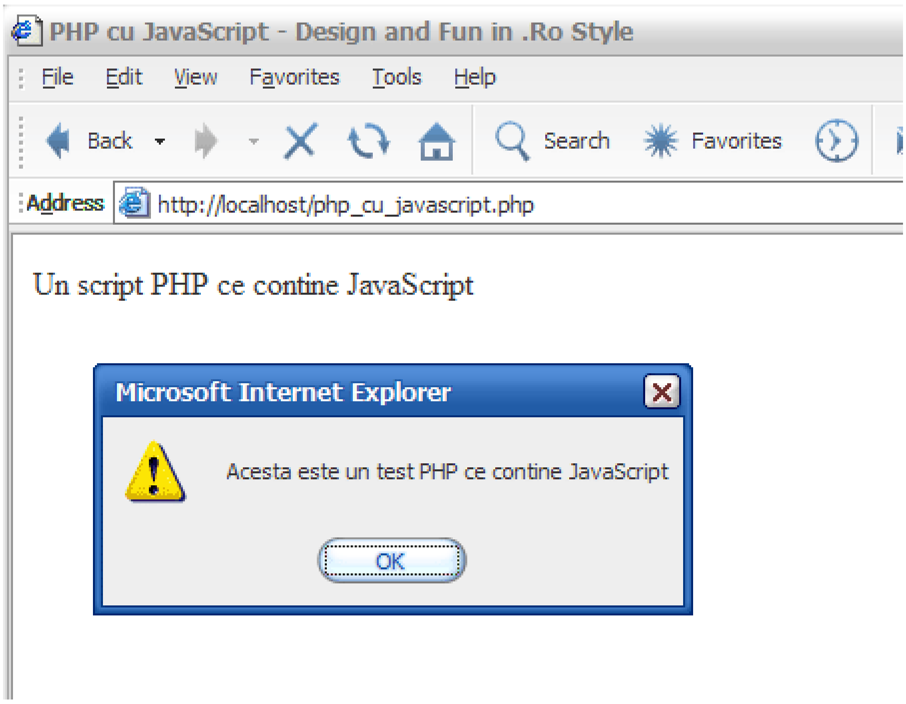
Aceste conflicte dintre ghilimelele simple magice si ghilimele simple din cod HTML sunt intalnite mai rar in HTML insa mare atentie atunci cand introduceti in scripturi PHP un cod **JavaScript** deoarece aici se folosesc mai des ghilimele simple.

```
<?php
echo '<html>
<head><title>PHP cu JavaScript</title>
<script language="JavaScript">
function alertare () {
        alert('Acesta este un test PHP ce contine JavaScript')
}
</script>
</head>
<body onLoad="alertare()">
<p>Un script PHP ce contine JavaScript</p>
</body>
</html>';
?>
```

Puneti codul intr-un fisier **php_cu_javascript.php**, salvati si apoi vizualizati in browser accesand http://localhost/php_cu_javascript.php



Vedeti aceeasi eroare ca cea explicata mai sus, efectuati modificarile, si anume din linia alert('Acesta este un test PHP ce contine JavaScript') schimbati in alert(\'Acesta este un test PHP ce contine JavaScript\') sau alert(“Acesta este un test PHP ce contine JavaScript”), apoi testati din nou in browser.



Observati faptul ca eroarea a disparut iar codul **JavaScript** a fost executat.

Nota: Toate exemplele pentru constructia echo si erorile aparute, au fost facute numai pentru constructia echo ce este delimitata de ghilimele magice simple, insa, dupa cum stiti / banuiti, erorile pot aparea si atunci cand constructia echo este delimitata de ghilimele magice duble, si va voi da un singur exemplu, iar restul de exemple le puteti aplica prin cele de mai sus.

Exemplu:

```
<?php
echo "Acesta este un exemplu de conflict cu ghilimele magice duble <br>
      <a href="pagina_mea.php">Legatura catre pagina mea</a>";
?>
```

Puneti codul intr-un fisier **conflict_ghilimele_duble.php**, salvati si apoi vizualizati in browser accesand http://localhost/conflict_ghilimele_duble.php

Apoi reparati eroarea si testati din nou in browser

```
<?php
echo "Acesta este un exemplu de conflict cu ghilimele magice duble <br>
      <a href=\"pagina_mea.php\">Legatura catre pagina mea</a>";
?>
```

sau

```
<?php
echo "Acesta este un exemplu de conflict cu ghilimele magice duble <br>
      <a href='pagina_mea.php'>Legatura catre pagina mea</a>";
?>
```

Cu speranta ca ati inteles constructia echo si ghilimelele magice, voi trece mai departe, insa nu inainte de a va da un sfat, si anume sa folositi tot timpul in constructiile voastre echo, ghilimele simple.

Este bine de stiut ca in scripturi PHP puteti anula bucati de cod sau puteti comenta linii din script.

De exemplu:

```
<?php
echo 'Un cod PHP comentat';  // Aceasta este un comentariu care nu se afiseaza in browser
// echo 'O linie din cod PHP care nu este afisata';
?>
```

Exista o alta posibilitate care se foloseste la inlaturarea temporara a unei bucati de cod PHP din pagina, si anume:

```
<?php
echo 'Aici este un text';
/*
echo 'Aici un altul';
echo 'Bine ai venit oriceon';
echo 'Un alt text';
*/
?>
```

Aceasta se foloseste pentru a nu fi nevoiti sa adaugam // la fiecare linie din script pentru a o anula.
Setam /* si la sfarsit */ si scapam mai usor de cod ☺

Variabile si lucrul cu acestea

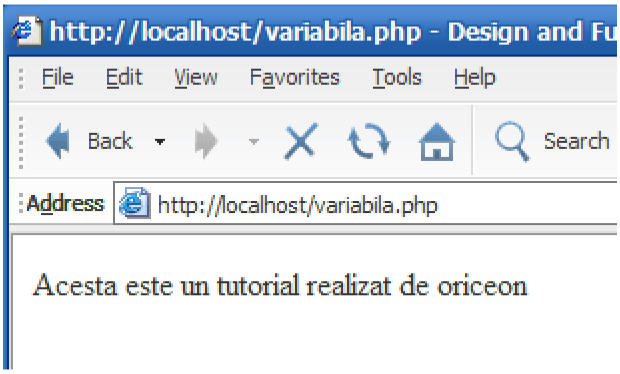
O variabila este o zona de memorie caruia i se da un nume pentru a putea fi recunoscuta ulterior si pentru a ne putea referi mai tarziu la ea. Ca sa avem o variabila, trebuie sa ii atribuim si o valoare.

```
<?php

$autor = 'oriceon';
echo 'Acesta este un tutorial realizat de '.$autor.' ';

?>
```

Puneti codul intr-un fisier **variabila.php**, salvati si apoi vizualizati in browser accesand http://localhost/variabila.php



Dupa cum vedeti, o variabila este construita dintr-un $ care se pune in fata ei, un = si ghilimele magice ' ' sau " " in interiorul carora se afla valoarea variabilei si apoi constructia este inchisa cu ;

In exemplul de mai sus avem declarata variabila cu numele autor si ca valoare textul oriceon

Nu uitati ca din constructia variabilei sa lipseasca cele de mai sus si anume $nume_variabila = ' '; sau $nume_variabila = " ";

Exemplul urmator contine o eroare, si intorcandu-ne din nou la cunostintele acumulate mai sus, in constructia **echo** si ghilimelele magice, vom incerca sa o rezolvam.

```
<?php

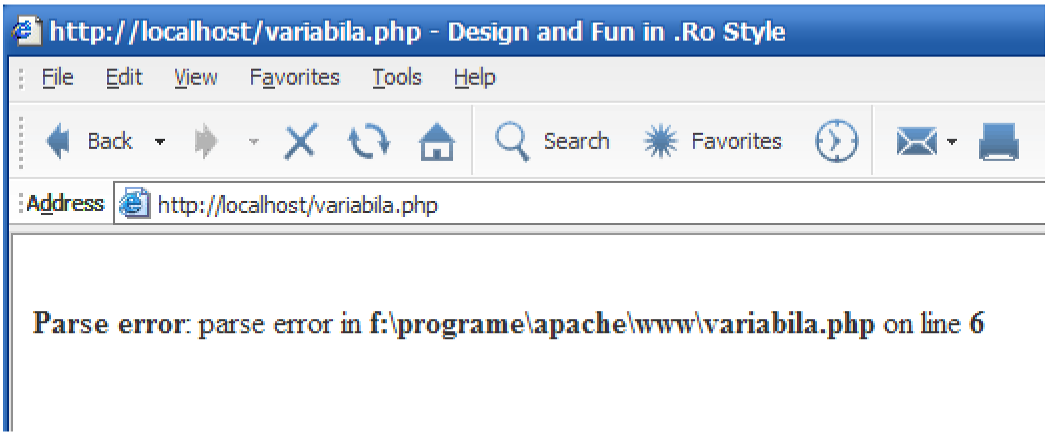
$autor = "oriceon';
echo 'Acesta este un tutorial realizat de '.$autor.' ';

?>
```

Codul acesta contine greseala la ghilimelele magice din constructia variabilei.
Vizualizand in browser, vom vedea eroarea de mai jos.

Modificati constructia $autor = "oriceon'; in $autor = 'oriceon'; apoi vizualizati in browser.

Veti observa ca problema a fost rezolvata iar codul PHP functioneaza corect.



Trecand mai departe la explicarea codului de mai sus, ajungem la partea unde afisam variabila pe pagina cu ajutorul constructiei **echo**.

```
echo 'Acesta este un tutorial realizat de '.$autor.' ';
```

Dupa cum vedeti este un echo in care ca noutate apare adaugarea variabilei cu ajutorul constructiei '..' si anume '.$autor.'. Aceasta constructie poate sa difere in functie de ghilimelele magice ale constructiei echo si anume:

1) Daca construtia echo este delimitata de ghilimele magice simple, adaugarea variabilei se face cu '..' si anume '.$autor.'
2) Daca constructia echo este delimitata de ghilimele magice duble, adaugarea variabilei se face cu ".." si anume ".$autor."

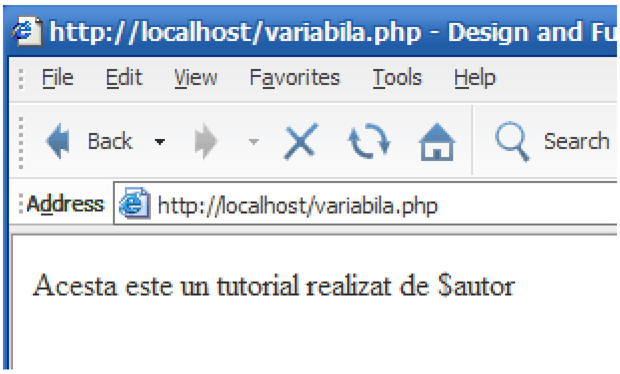
Aceeasi modalitate de adaugare o sa se foloseasca si pentru **$_SESSION** , **$_POST** , **$_GET** si alte variabile predefinite pe care le vom invata mai tarziu.

Daca variabila nu este adaugata corect, in sensul ca nu i se atribuie constructia corespunzatoare adaugarii, vom avea urmatoarea afisare in browser pentru codul urmator:

```
<?php

$autor = 'oriceon';
echo 'Acesta este un tutorial realizat de $autor ';

?>
```

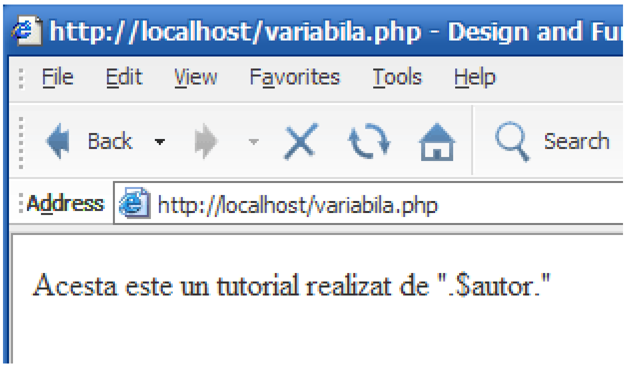


Uitati-va la constructia afisarii variabilei $autor si observati ca nu este corecta deoarece nu are '..'.

Modificam codul din nou si o sa vedem o noua afisare gresita

```
<?php
$autor = 'oriceon';

echo 'Acesta este un tutorial realizat de ".$autor." ';
?>
```



Iar, intorcandu-ne la ghilimelele magice si constructia afisarii variabilei pe pagina, o sa ne aducem aminte unde am gresit.

In caz ca nu gasiti raspunsul la aceasta eroare, cititi cu o pagina mai sus.

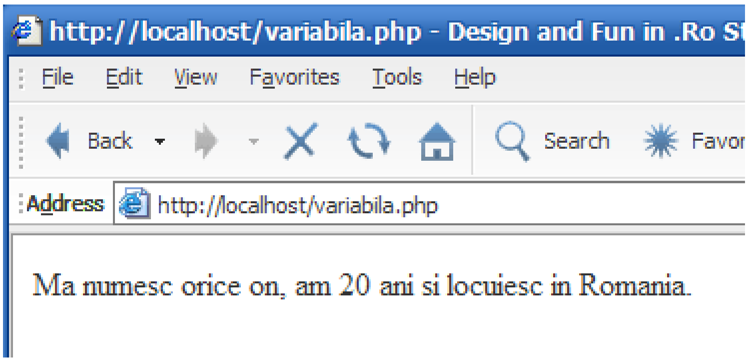
Nota: Intr-un script php, puteti avea cate variabile doriti.

Exemplu:

```
<?php
$nume = 'orice';
$prenume = 'on';
$varsta = '20 ani';
$tara = 'Romania';

echo ' Ma numesc  '.$nume.' '.$prenume.', am '.$varsta.' si locuiesc in '.$tara.'. ';
?>
```

Vizualizati in browser si rezultatul o sa fie:



Intr-o variabila puteti avea mai multe variabile cu continut diferit iar “lipirea” variabilelor se face cu .’ si respectiv ‘. si anume daca avem 2 variabile le vom uni astfel $variabila1.’ ‘.$variabila2

```
<?php
$nume = 'orice';
$prenume = 'on';
$nume_complet = $nume.' '.$prenume;
echo ' Ma numesc  '.$nume_complet.' ';
?>
```

Daca doriti ca o variabila sa aibe ca valoare un text definit direct in ea si sa mai fie alaturat acelui text.. un altul dintr-o alta variabila, puteti folosi urmatorul cod PHP:

```php
<?php
$nume = 'orice';
$prenume = 'on';
$nume_complet = 'Numele meu complet este '.$nume.' '.$prenume;
echo ' '.$nume_complet.' ';
?>
```

O singura variabila se poate afisa in browser si fara ajutorul ghilimelelor magice, si anume:

```php
<?php
$nume = 'orice';
$prenume = 'on';
$nume_complet = 'Numele meu complet este '.$nume.' '.$prenume;
echo $nume_complet;
?>
```

Constante

O constanta stocheaza o valoare, cum este si o variabila, dar aceasta valoare, dupa ce a fost stabilita, nu mai poate fi modificata in script.

Pentru a defini o constanta, ne vom folosi de functia **define();** iar numele constantelor este scris cu MAJUSCULE, aceasta optiune nu este obligatorie, insa va face codul dumneavoastra mai frumos si mai lizibil.

O diferenta importanta intre constante si variabile, este faptul ca o constanta nu are in fata ei semnul $.

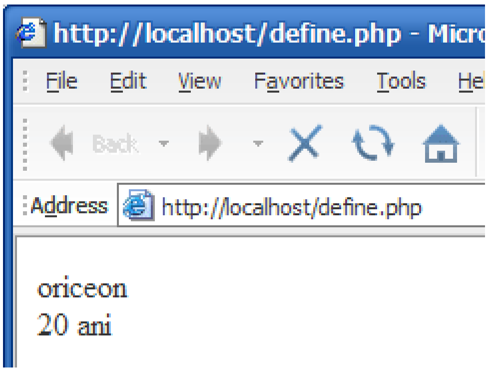
```
<?php

define('AUTOR', 'oriceon');
define('VARSTA', '20 ani');

echo AUTOR.' <br> '.VARSTA;

?>
```

Realizati o pagina cu numele **define.php** si apoi testati in browser pentru a observa rezultatele.



Variabile predefinite

$GLOBALS = pot fi accesate toate variabilele globale care sunt accesibile script-ului PHP curent
$_SERVER = contine o serie de variabile ale caror valori sunt setate de server-ul web; majoritatea valorilor variabilelor din acest vector depind de mediul de executie al script-ului curent.
$_GET si $_POST contin variabile primite de script prin intermediul unor transferuri care folosesc metodele HTTP get, respectiv post. De exemplu, prin intermediul acestor vectori, pot fi accesate valorile campurilor dintr-un formular care a fost completat si transmis folosind una dintre cele doua metode.
$_COOKIE contine valorile variabilelor care cuprind informatii referitoare la cookie-urile pastrate pe calculatorul utilizatorului ce acceseaza pagina web.
$_FILES contine variabile primite de script prin intermediul incarcarilor de fisiere prin metoda *post*.
$_ENV contine variabile disponibile prin intermediul mediului in care este executat.
$_REQUEST contine variabile disponibile prin intermediul oricarui tip de mecanism cu ajutorul caruia utilizatorul poate introduce date.
$_SESSION contine variabile care corespund sesiunii curente a script-ului.

Tipuri de variabile

Tipul unei variabile se refera la genul de date care sunt introduse in ea.

Variabilele pot fi de mai multe tipuri, nu doar numere. PHP are opt tipuri de variabile. Patru dintre acestea sunt tipuri scalare (*boolean, integer, float* si *string*), doua sunt tipuri compuse (*array* si *object*), iar alte doua sunt tipuri speciale (*resource* si *null*). De asemenea, din motive de lizibilitate, au fost introduse trei pseudotipuri: *mixed, number* si *callback*. Mai exista si tipul *double*, dar semnificatia acestuia este aceeasi cu cea a tipului *float*. Cele doua denumiri coexista doar din motive "istorice". In PHP, de obicei, tipul unei variabile nu este specificat de catre programator, ci este stabilit in timpul executiei in functie de contextul in care este folosita variabila.

Tipul boolean:

Variabilele de acest tip pot avea doar doua valori: *ADEVARAT* sau *FALS*.

Aceste valori pot fi indicate prin cuvintele cheie *TRUE* sau *FALSE* (pentru ambele nu se face distinctie intre literele mari si literele mici). Exista posibilitatea de a converti o variabila de orice tip la tipul *boolean.* In momentul efectuarii unei conversii, sunt convertite la valoarea *FALSE* urmatoarele valori:

- numarul intreg 0;
- numarul real 0.0;
- sirul vid;
- sirul "0";
- un vector fara nici un element;
- un obiect fara nici o variabila membru;
- o variabila de tipul *NULL*;
- o variabila nedefinita.

Orice alta valoare este convertita la valoarea *TRUE* (inclusiv resursele).

Acest tip se poate folosi de exemplu pentru verificarea logarii intr-o pagina de administrare. Dupa ce se fac verificarile, daca utilizatorul este logat ca administrator, functia noastra va returna o valoare de adevar: TRUE daca este logat sau FALSE daca nu este, si astfel vom sti daca sa ii acordam sau nu acces in sectiunea de administrare.

Tipul integer:

O variabila de tip *integer* reprezinta o valoare din multimea numerelor intregi.

Aceste numere pot fi specificate in baza 10, in baza 16 sau in baza 8, conventiile fiind aceleasi ca si in limbajele C/C++ sau Java. Modul de reprezentare depinde de platforma utilizata; de obicei se foloseste reprezentarea pe 32 de biti.

Interpretorul PHP nu ofera suport pentru numerele intregi fara semn. Trebuie remarcat faptul ca in PHP nu exista nici un operator pentru efectuarea de impartiri intregi.
De exemplu, rezultatul operatiei 3/2 nu va fi numarul intreg 1 (ca in C/C++ sau Java), ci numarul real (*float*) 1.5

Si pentru numerele intregi exista posibilitatea efectuarii de conversii:
- valoarea logica *TRUE* este convertita la valoarea intreaga 1;
- valoarea logica *FALSE* este convertita la valoarea intreaga 0;

- un numar real este convertit prin "rotunjire inspre 0"; asadar, valoarea reala 2.5 va fi convertita la valoarea intreaga 2, in timp ce valoarea reala -2.5 va fi convertita la valoarea intreaga -2;
- un sir de caractere este convertit luand in considerare doar primele caractere care contin informatii numerice; asadar sirul "10" va fi convertit la valoarea intreaga 10; de asemenea sirul "10 ani" va fi convertit tot la valoarea 10; daca primele caractere nu contin informatii numerice, rezultatul conversiei va fi valoarea 0.

Tipul float:

O variabila de tip *float* poate fi specificata folosind fie forma zecimala, fie cea stiintifica (cu exponent).

La fel ca si in cazul tipului *integer*, precizia variabilelor de tipul float este dependenta de platforma utilizata. De obicei se foloseste standardul IEEE 64. Exista posibilitatea de a converti o variabila de orice tip la tipul *float*.

Pentru numerele reale se pot efectua urmatoarele conversii:

- un sir de caractere este convertit luand in considerare doar primele caractere care contin informatii numerice; asadar sirul "10.2" va fi convertit la valoarea reala 10.2; sirul "1.23E1 ani" va fi convertit la valoarea 12.3;
- in toate celelalte cazuri se realizeaza conversii la numere intregi care apoi sunt convertite la valorile reale corespunzatoare.

Tipul string:

O variabila de tip *string* reprezinta un sir de caractere.

Un caracter se reprezinta pe un octet, deci sunt 256 de caractere distincte. Acest lucru implica faptul ca interpretorul PHP nu ofera suport nativ pentru setul de caractere *Unicode*. Lungimea variabilelor de tip *string* nu este limitata de catre interpretor.

Literalii de tip sir de caractere pot fi specificati in trei moduri diferite:

- prin folosirea ghilimelelor simple (exemplu $a='acesta este un sir de caractere'). Pentru a avea in cadrul sirului simbolul "`", atunci inaintea acestuia trebuie scris caracterul "\", iar pentru a putea specifica simbolul "\" acesta trebuie dublat.
- prin folosirea ghilimelelor duble. Folosind aceasta notatie, pot fi specificate mai multe caractere speciale, pe langa caracterele de la varianta anterioara, printre care: sfarsit de linie ("\r"), rand nou ("\n"), tab orizontal ("\t"), semnul dolar ("\$"), ghilimelele duble ("\""), secvente de caractere pentru specificarea faptului ca o expresie regulara este in notatie octala ("\[0-7]{1,3}") si secventele de caractere pentru specificarea faptului ca o expresie regulara este in notatie hexazecimala ("\x[0-9A-Fa-f]{1,2}"). Cel mai important lucru este acela ca, folosind acest mod de specificare a literalilor de acest tip, numerele de variabile care apar in interior vor fi transformate in valoarea lor. De exemplu, daca $a este o variabila de tipul *integer* si are valoarea 2, atunci sirul de caractere "Variabila a are valoarea $a." va fi transformat in sirul "Variabila a are valoarea 2".
- notatia *heredoc*. Acest tip de notatie a fost introdus la versiunea 4 a interpretorului PHP. Pentru a specifica un sir de caractere folosind aceasta notatie, trebuie utilizat operatorul "<<<" urmat de un identificator ales de utilizator. Toate caracterele care se afla intre operatorul "<<<", urmat de un identificator pe o singura linie, si acelasi identificator pe o alta linie, vor constitui valoarea sirului de caractere. De exemplu, instructiunea:

$str=<<<SF

Acesta este un exemplu de utilizare a sintaxei heredoc SF; va avea ca rezultat un sir de caractere format din trei linii de text.

Pentru a accesa un anumit caracter din sirul de caractere, se foloseste, dupa numele variabilei de tip string,

indicile caracterului care trebuie accesat scris intre acolade. De exemplu, $str{0} returneaza primul caracter din sirul de caractere $str.

In cazul in care dorim sa concaternam doua siruri de caractere, vom folosi operatorul "**.**" Folosirea operatorului "+" nu va concaterna cele doua siruri.
Exista posibilitatea de a converti o variabila de orice tip la tipul *string.* Pentru sirurile de caractere, se pot efectua urmatoarele conversii:

- valoarea logica TRUE va fi convertita la sirul "1", iar valoarea logica FALSE va fi convertita la sirul vid ("");
- un numar intreg va fi convertit la un sir de caractere care reprezinta valoarea numarului in baza 10;
- un numar real va fi convertit la un sir de caractere care reprezinta notatia stiintifica a acestuia;
- obiectele sunt intotdeauna convertite la sirul "*Object*";
- variabilele de tipul *resource* sunt convertite la sirul "*Resource id #n*", unde n reprezinta un numar unic atasat resursei respective de catre interpretorul PHP;
- valoarea *NULL* este convertita la sirul vid ("").

Tipul array:

Vectorii in PHP sunt niste multimi formate din *chei*. Fiecarei chei din vector i se ataseaza o valoare.
Acest tip de date este optimizat astfel incat sa poata fi folosit in locul urmatoarelor structuri de date: liste, tabele de dispersie, dictionare, colectii, stive, cozi si altele.

Datorita faptului ca o valoare poate fi reprezentata de un alt vector, se pot simula foarte usor arborii *n*-dimensionali sau tablourile *n*-dimensionale. Valoarea unei variabile de tip vector se poate specifica folosind constructia array (cheie => valoare, cheie => valoare, ...)

De exemplu, urmatoarea instructiune PHP va construi un vector cu doua elemente, dintre care unul este de tip *string*, iar celalalt de tip *boolean*:

```
$a = array('ch' => 'string', 12 => TRUE);
```

Variabila $a reprezinta un ector, $a["ch"] are valoarea *string*, iar $a[12] are valoarea *TRUE.*
In cazul in care nu se specifica o cheie pentru o valoare, atunci acea valoare va fi atasata unei chei care va fi cheia maxima de tip *integer* folosita anterior, la care se adauga valoarea 1. Cheile pot avea si valori negative. Daca nu exista chei de tip *integer*, atunci valoarea va fi atasata cheii 0. De exemplu, urmatoarele doua instructiuni sunt echivalente:

```
array(5 => 43, 32, 56, 'b' => 12);
array(5 => 43, 6 => 32, 7 => 56, 'b' => 12);
```

Daca se foloseste valoarea logica *TRUE* ca si cheie, atunci aceasta va fi convertita la cheia de tip intreg 1, iar valoarea *FALSE* va fi convertita la numarul intreg 0. Nu se pot folosi pentru chei variabile de tipul *array* sau *object*.

O variabila de tip array se poate modifica prin setarea explicita de valori.
De exemplu instructiunea $a["x"] = 42; adauga in vectorul $a valoarea 42 atasata cheii "x".
Daca se foloseste un vector care nu a fost definit anterior, atunci acesta este creat automat.
Asadar printr-o instructiune de forma $a[5] = 42, in cazul in care vectorul $a nu exista, atunci se va crea un vector cu un singur element. Cheia acestuia va fi numarul intreg 5, iar valoarea sa va fi 42.

De asemenea, exista posibilitatea de a crea un element nou fara a-i preciza cheia. Sintaxa are forma $vector[] = valoare; aceasta instructiune are ca efect adaugarea unui element a carui cheie este un numar intreg mai mare cu 1 decat cel mai mare numar intreg care este cheie a unui alt element al vectorului.

Daca nu exista nici o astfel de cheie, atunci noul element va avea cheia 0.

De exemplu, urmatoarele doua secvente sunt echivalente:

$a[5] = 1; .... $a[5] = 1;
$a[6] = 2; .... $a[] = 2;

Prin conversia la un vector a unei variabile de tip scalar *(boolean, integer, float, string*) sau *resource* se creeaza un vector cu un singur element; cheia acestui element este numarul intreg 0, iar valoarea este cea a variabilei convertite.

Daca se converteste un obiect (variabila de tip object), atunci vectorul rezultat va contine cate un element pentru fiecare variabila membru a obiectului. Cheile elementelor vor fi date de denumirile proprietatilor obiectului (variabilele membru ale obiectului), iar valorile elementelor vor fi valorile proprietatilor obiectului. Daca realizam o conversie a unei variabile de tip NULL, atunci rezultatul va fi un vector vid (care nu contine nici un element).

In continuare sunt prezentate cateva exemple care descriu mai detaliat posibilitatile oferite de folosirea vectorilor in PHP.

Pentru inceput, prezint un vector al carui elemente reprezinta caracteristicile unei portocale:

```
<?php

$a = array ('denumire' => 'portocala',
            'familie' => 'citrice',
            'culoare' => 'portocaliu',
            'forma' => 'rotunda',
            'gust' => 'dulce'
);

echo $a['denumire'].' <br> '.$a['familie'];

?>
```

Putem adauga si alte elemente care sa reprezinte diferite alte proprietati. De exemplu, am putea avea nevoie de o valoare suplimentara careia nu dorim sa ii atribuim nici un nume de identificare (cheie). Pentru ca vectorul sa contina un element suplimentar cu valoarea 4, vom putea defini vectorul astfel:

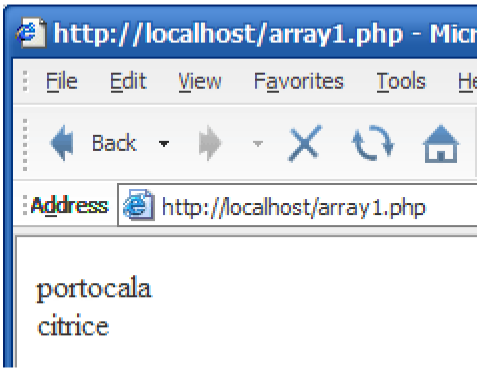
```
<?php

$v = array ('denumire' => 'portocala',
            'familie' => 'citrice',
            'culoare' => 'portocaliu',
            'forma' => 'rotunda',
            'gust' => 'dulce',
             4
);

echo $v['denumire'].' <br> '.$v['familie'];

?>
```

Creati o pagina cu numele array1.php apoi testati in browser.



Cheia elementului cu valoarea 4 va fi numarul intreg 0 deoarece nu exista nici o alta cheie care este numar intreg. O alternativa de construire a acestui vector este urmatoarea:

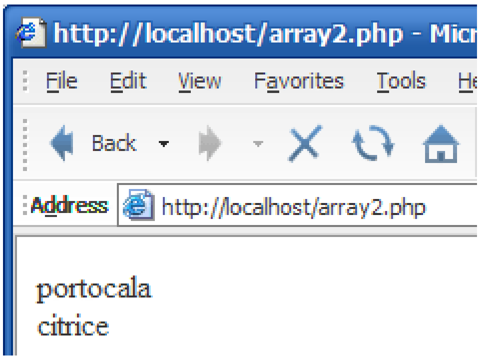
```php
<?php

$v['denumire'] = 'portocala';
$v['familie'] = 'citrice';
$v['culoare'] = 'portocaliu';
$v['forma'] = 'rotunda';
$v['gust'] = 'dulce';
$v[] = 4;

echo $v['denumire'].' <br> '.$v['familie'];

?>
```

Creati o alta pagina cu numele **array2.php**, introduceti codul de mai sus, apoi comparati cu rezultatul si codul din pagina **array1.php**.



Efectuati cateva exercitii cu array-uri pentru aprofundare.

Folosirea unor array-uri ce au in alcatuire alte array-uri.

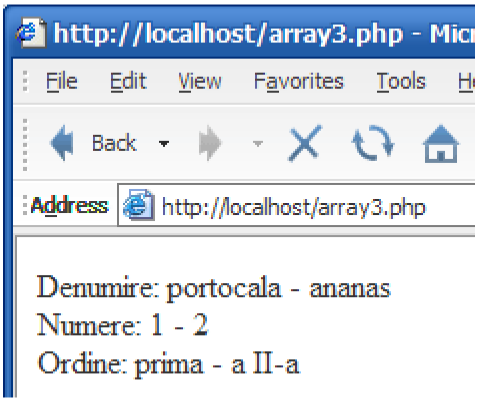
```php
<?php

$fructe = array (
    'denumire'  => array('p' => 'portocala', 'a' => 'ananas', 'm' => 'mar'),
    'numere'    => array(1, 2, 3, 4, 5, 6),
    'ordine'    => array('prima', 5 => 'a II-a', 'a III-a')
);

echo 'Denumire: '.$fructe['denumire']['p'].' - '.$fructe['denumire']['a'].' <br>';
echo 'Numere: '.$fructe['numere'][0].' - '.$fructe['numere'][1].' <br>';
echo 'Ordine: '.$fructe['ordine'][0].' - '.$fructe['ordine'][5].' <br>';

?>
```

Creati o pagina cu numele **array3.php**, introduceti codul de mai sus, apoi testati in browser.



Tipul object:

Pentru a defini un obiect care poate fi folosit pentru afisarea mesajului “Salutare lume!”, se scrie urmatoarea secventa:

```
<?php

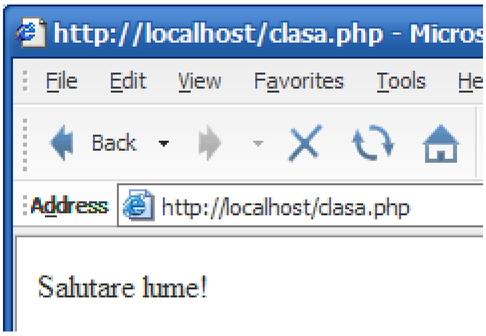
class Salutare {
  function ArataSalutare() {
  echo 'Salutare lume!';
  }
}

$a = new Salutare;

echo $a -> ArataSalutare();

?>
```

Creati o pagina cu numele **clasa.php**, introduceti codul de mai sus apoi testati in browser.



Observam faptul ca ne este afisat textul “Salutare lume!”, text ce l-am introdus in functia ArataSalutare()

Pentru a utiliza o variabila de tip obiect, va trebui sa realizam o *instantiere* prin intermediul instructiunii *new*.

Sintaxa este:

```
$a = new Salutare;
```

Astfel, variabila $a devine un obiect ale carui metode pot fi utilizate. Pentru afisarea propriu-zisa a mesajului va trebui sa executam metoda *Displaysalutare()* printr-o instructiune de tipul $a -> Displaysalutare();

Orice variabila de un anumit tip poate fi convertita intr-un obiect.

Daca variabila respectiva este un obiect, atunci ea nu va fi modificata. In caz contrar, efectul conversiei este crearea unei noi instante a clasei *stdClass*.

Daca variabila are tipul *NULL*, atunci noua instanta va fi vida. In toate celelalte cazuri, instanta va contine o variabila membru numita scalar a carei valoare va fi cea a variabilei convertite. Pentru conversii vom folosi instructiuni de tipul $obiect = (object) "Salutare lume!". Dupa realizarea conversiei, vom putea tipari mesajul “Salutare lume!” folosind instructiunea echo $obiect->scalar;

Tipul resource:

Variabilele de tip *resource* sunt folosite pentru pastrarea unor referinte catre anumite resurse externe cum ar fi conexiuni la baze de date, fisiere, etc. Resursele sunt create si utilizate de anumite functii speciale.

Datorita specificului acestui tip de date, valoarea nici unei variabile de alt tip nu poate fi convertita la tipul *resource*.

Tipul NULL:

Valoarea speciala *NULL* este atribuita oricarei variabile care nu a fost initializata. Aceasta valoare este singura pe care o pot avea variabilele de tip *NULL*.

Se considera ca o variabila are tipul *NULL* daca:

- i s-a atribuit constanta NULL;
- nu a fost initializata;
- a fost dezinitializata (prin intermediul functiei unset ()).

Siruri si caractere speciale

Spre deosebire de intregi si de numere duble, care contin cu precadere cifre, sirurile pot contine orice caracter.

Ca atare, sirurile sunt utile pentru stocarea datelor care nu pot fi calculate, precum nume si adrese. De asemenea, sirurile pot fi utilizate pentru stocarea datelor numerice. Pentru a specifica un sir in PHP, caracterele care alcatuiesc sirul sunt incluse NUMAI intre ghilimele duble **"** si **"**

De exemplu, sirul reprezentand numele "Ivascu Valentin".

Asa cum am mai spus, un sir poate contine date numerice; de exemplu , "3.141516".
PHP faciliteaza includerea in siruri a unor caractere speciale, precum caracterele de salt la linie noua.

\n - salt la linie noua
\r - retur de car
\t - caracter de tabulare pe orizontala
\\ - backslash
\$ - simbolul dolarului
\" - ghilimele duble

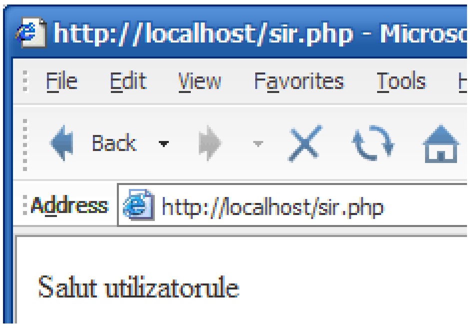
Ca exemplu, iata un sir care include un retur de car, urmat de un salt la linie noua:

```
<?php

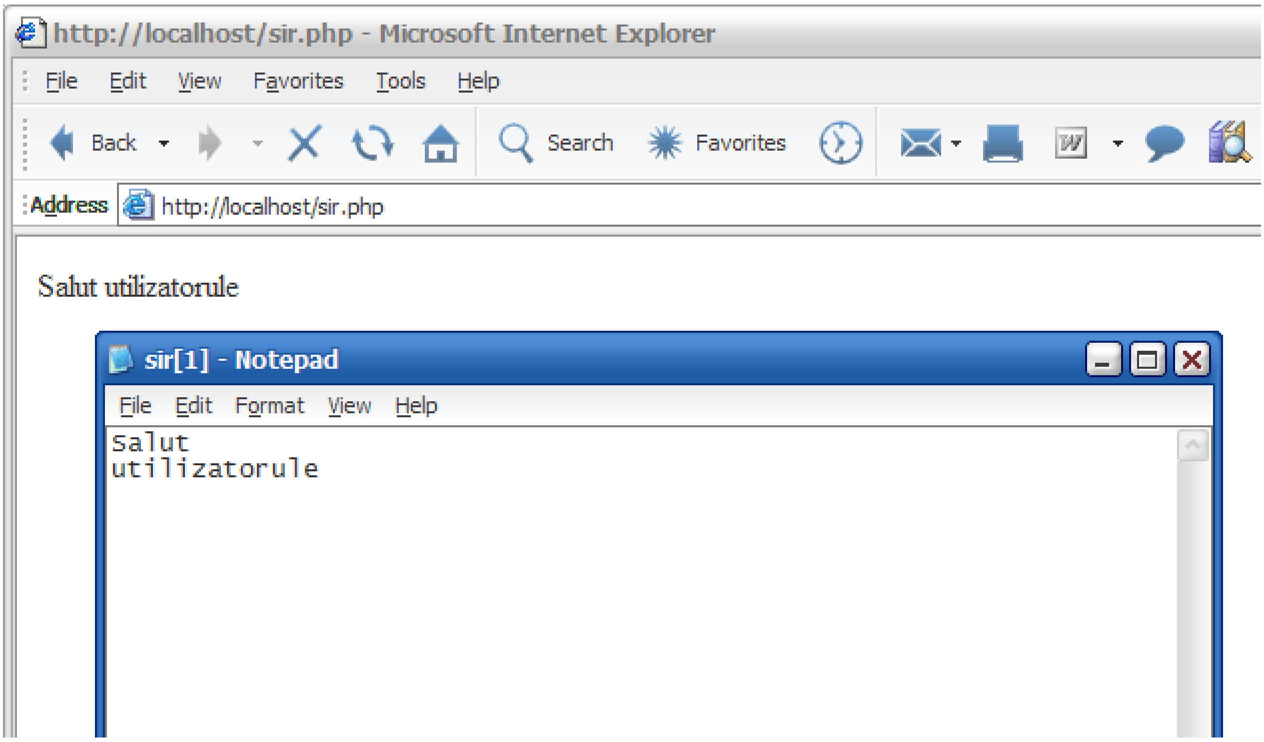
echo "Salut \r\nutilizatorule";

?>
```

Creati o pagina cu numele **sir.php**, introduceti codul de mai sus apoi testati in browser.



Apasati pe view (sus in browser) apoi source – pentru a vizualiza sursa paginii sir – si observati ca textul “utilizatorule” apare pe o noua linie in codul din sursa paginii.



Retineti ca fiecare secventa escape incepe cu un backslash (\). Pentru a include un backslash intr-un sir, trebuie sa folositi secventa escape adecvata, care este alcatuita din doua caractere backslash. Pentru a introduce ghilimele duble in cadrul unui sir fara a folosi secventa escape , puteti include sirul intre ghilimele simple astfel: 'Pe ea o cheama "Nikita"'

Clase si obiecte

Ce este o clasa ?

O clasa este o colectie de variabile si functii care opereaza asupra variabilelor respective.

Sintaxa folosita pentru declararea unei clase in PHP este:

```
<?php

class nume_clasa {
// date membre
var nume_variabila_1
// ...
var nume_variabila_m*
// metode
function nume_functie_1 (parametri) {
// definitia functiei
}
// ...
function nume_functie_n (parametri) {
// definirea functiei
}
}

?>
```

Pentru numele unei clase poate fi utilizat orice identificator permis in PHP cu o singura exceptie: sdtclass

Acest identificator este folosit de PHP in scopuri interne. In PHP functiile ale caror identificatori incep cu '_' sunt considerate functii magice si utilizarea acestora nu este recomandata. In PHP, datele membre nu pot fi initializate decat cu valori constante. Pentru a initializa variabilele cu valori care nu sunt constante trebuie folosit un constructor.

Mai jos aveti un exemplu de clasa in care initializarile **nu sunt corecte**:

```
class Nepermis {
var $data = date ('Y-m-d');
var $nume = $prenume;
var $dest = 'Ivascu' . 'Valentin';
var $obiecte = array ('orice', 'on');
}
```

Obiectele

In PHP clasele sunt considerate a fi tipuri de date; ele pot fi privite ca fiind "amprentele" variabilelor propriu-zise. Pentru a crea o variabila al carei tip este o clasa, trebuie utilizat operatorul **new**. In continuare, vom defini o clasa Aritmetica cu doua date membre x si y care sunt numere intregi si doua metode care realizeaza adunarea, respectiv inmultirea lor.

```
class Aritmetica {
var x = 2;
var y = 3;
  function Suma() {
  return $this -> x + $this -> y;
  }
     function Produs() {
     return $this -> x * $this -> y;
     }
}
```

Pentru a crea un obiect de tipul Aritmetica, vom utiliza o instructiune de tipul:

$aritm = new Aritmetica;

Acum putem utiliza metodele clasei; pentru a afisa suma sau produsul celor doua numere, vom putea apela cele doua metode astfel:

```
echo $aritm -> Suma();
echo $aritm -> Produs();
```

Vom obtine rezultatele 5, respectiv 6. Valorile datelor membre pot fi si ele modificate prin instructiuni de tipul:

```
$aritm -> x = 5;
$aritm -> y = 4;
```

Daca, in urma modificarii apelam din nou metodele Suma 9 si Produs(), rezultatele vor fi 9, respectiv 20.

Haideti sa realizam o clasa si sa vedem cum lucreaza.

```
<?php

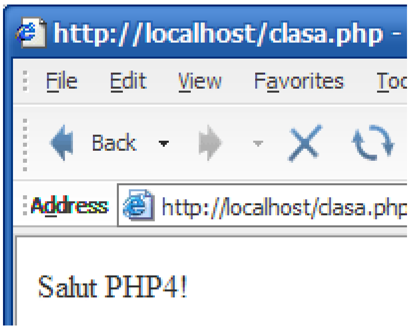
class PHP4 {
var $salut = 'Salut PHP4!';
    function Salut() {
    return $this -> salut;
    }
}

$salutare = new PHP4;

echo $salutare -> Salut().'<br>';

?>
```

Realizati o pagina cu numele **clasa.php**, puneti codul de mai sus, apoi testati in browser.



In acest exemplu a fost utilizata pseudo-variabila $this.
Aceasta este folosita pentru a indica faptul ca se opereaza asupra unei date membre a obiectului curent.

Extinderea claselor

Deseori este necesara definirea unor clase cu proprietati (date membre) si metode asemanatoare.
Pentru a usura definirea unor astfel de clase, a fost introdus conceptul de **extindere** (derivare) a claselor.

O clasa derivata va pastra toate proprietatile si metodele clasei pe care o extinde si poate contine diferite proprietati si metode noi. Nu exista nici o posibilitate de a elimina din clasa derivata anumite proprietati sau metode ale clasei de baza.

O anumita clasa poate avea o singura clasa **parinte**; asadar, in PHP nu este permisa **mostenirea** multipla. Pentru a extinde o anumita clasa se utilizeaza cuvantul cheie **extends**. In urmatorul exemplu, vom extinde clasa *Aritmetica*; vom adauga inca o variabila si vom crea doua noi functii: una pentru calculul sumei celor trei variabile si una pentru calcularea produsului lor:

```
class Aritmetica3 extends Aritmetica {
var z = 4;
   function Suma3() {
   return $this -> x + $this -> y + $this -> z;
   }
      function Produs3() {
      return $this -> x * $this -> y * $this -> z;
      }
}
```

Daca definim un obiect prin intermediul unei instructiuni de genul:

```
$aritm3 = new Aritmetica3
```

atunci pentru acest obiect vom putea utiliza atat metodele definite in cadrul clasei *Aritmetica3: Suma3()* si *Produs3()*, cat si metodele definite in cadrul clasei de baza *Aritmetica: Suma()* si *Produs()*. In continuare aveti un exemplu care ilustreaza modul in care pot fi create si utilizate clasele derivate.

In exemplul urmator veti observa flexibilitatea claselor.
Creati o pagina cu numele **clasa2.php**, introduceti codul urmator apoi testati in browser pentru a observa rezultatele.

```php
<?php

class Aritmetica {
var $x = 2;
var $y = 3;
    function Suma() {
    return $this -> x + $this -> y;
    }
       function Produs() {
       return $this -> x * $this -> y;
       }
}
class Aritmetica3 extends Aritmetica {
var $z = 4;
    function Suma3() {
    return $this -> x + $this -> y + $this -> z;
    }
       function Produs3() {
       return $this -> x * $this -> y * $this -> z;
       }
}

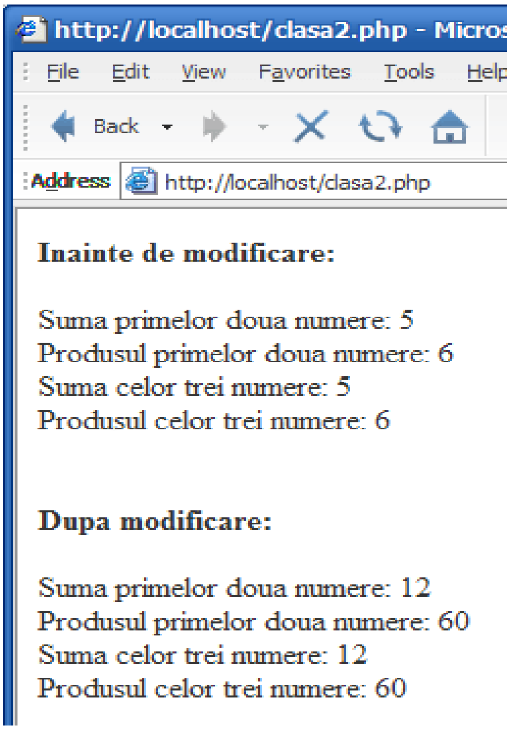
$aritm3 = new Aritmetica3;

echo '<b>Inainte de modificare:</b> <br><br>
         Suma primelor doua numere: '.$aritm3 -> Suma().' <br>
         Produsul primelor doua numere: '.$aritm3 -> Produs().' <br>
         Suma celor trei numere: '.$aritm3 ->Suma().' <br>
         Produsul celor trei numere: '.$aritm3 -> Produs().' <br><br><br>';

$aritm3 -> x = 5;
$aritm3 -> y = 4;
$aritm3 -> z = 3;

echo '<b>Dupa modificare:</b><br><br>
         Suma primelor doua numere: '.$aritm3 -> Suma3().' <br>
         Produsul primelor doua numere: '.$aritm3 -> Produs3().' <br>
         Suma celor trei numere: '.$aritm3 -> Suma3().' <br>
         Produsul celor trei numere: '.$aritm3 -> Produs3().' <br>';

?>
```



In PHP clasele trebuie definite inaintea utilizarii lor; asadar clasa parinte va fi definita intotdeauna inaintea clasei **fiu**.
Relatia dintr-o clasa derivata si clasa pe care o extinde poarta denumirea de relatie parinte-fiu.

Constructori

Un constructor este o metoda (functie) a unei clase care este apelata automat in momentul in care este creata o noua instanta a clasei (cu ajutorul operatorului **new**).

In PHP, considerata ca fiind un constructor, orice functie care are acelasi nume cu clasa in interiorul careia este definita. Constructorii pot fi folositi pentru initializarea datelor membre cu valori care nu sunt constante. Ei pot avea argumente, iar acestea pot fi optionale. Pentru a putea utiliza clasa fara a specifica nici un parametru in momentul crearii unui obiect, se recomanda stabilirea unor valori implicite pentru toate argumentele constructorului. In cazul in care nu este definit un constructor pentru o anumita clasa, se utilizeaza constructorul clasei de baza, daca aceasta exista.

De exemplu, pentru urmatoarea secventa de cod, in momentul crearii obiectului corespunzator variabilei $b, va fi apelat constructorul clasei A.

```
class A {
    function A() {
    echo 'Constructorul clasei A. <br>';
    }
       function B() {
       echo 'O functie obisnuita a clasei A. <br>';
       }
}

class B extends A {
      function C() {
      echo 'O functie obisnuita a clasei B. <br>';
      }
}

$b = new B;
```

In PHP apelul constructorului clasei de baza, trebuie sa fie explicit daca este necesara executarea operatiilor corespunzatoare. In majoritatea limbajelor de programare exista functii speciale numite destructori care sunt apelate automat in momentul "distrugerii" unui obiect. In PHP nu exista destructori.

Operatorul ::

Uneori este utila folosirea unor metode sau variabile ale clasei de baza sau ale unei clase care nu a fost instantiata inca. In acest scop, a fost introdus operatorul **::**.

Pentru a descrie modul de utilizare al acestui operator, vom prezenta mai intai un exemplu:

```
class A {
    function exemplu() {
    echo 'Functia clasei de baza. <br>';
    }
}

class B extends A {
    function exemplu() {
    echo "Functia redefinita<br>\n";
    A :: exemplu();
    }
}

A :: exemplu();

$b = new B;
$b -> exemplu();
```

Prin intermediul instructiunii *A :: exemplu();* este apelata metoda *exemplu()* a clasei A, asadar se afiseaza mesajul '*Functia clasei de baza*' cu toate ca nu exista nici un obiect care este o instanta a acestei clase, deci nu putem scrie o instructiune de tipul *$a -> exemplu();* In schimb apelam metoda *$b -> exemplu();* ca "o functie a clasei" si nu ca "o functie a unui obiect".

Putem avea functii ale claselor, dar nu putem avea variabile ale claselor. De fapt, in momentul unui astfel de apel, nu se creeaza nici un obiect care este instanta a clasei respective. Ca urmare, o functie a unei clase nu poate opera asupra unor proprietati ale clasei, dar poate utiliza variabile locale sau globale. In plus, o astfel de functie nu poate utiliza pseudo-variabila **$this**. In exemplul anterior, in cadrul clasei B este redefinita functia *exemplu()*. Asadar, definitia "originala" (din cadrul clasei A) nu poate fi accesata in interiorul clasei B decat daca ne referim la ea explicit prin intermediul operatorului **::**.

Accesarea clasei de baza

In exemplul anterior am utilizat o functie a clasei de baza.
In locul utilizarii denumirii clasei de baza poate fi folosita denumirea speciala parent care este o referinta la clasa de baza definita in cadrul constructiei extends.

Folosirea denumirii speciale este utila in cazul in care ierarhia de clase se modifica. In acest caz este suficienta o singura modificare in cadrul constructiei extends, fara a mai fi necesare modificari in interiorul clasei derivate.

Asadar, definitia clasei B poate fi rescrisa astfel:

```
class B extends A {
      function exemplu() {
      echo "Functia redefinita<br>\n";
      parent :: exemplu();
      }
}
```

Serializarea obiectelor

Prin serializare se intelege crearea unui sir de octeti care contine reprezentarea interna (binara) a variabilei respective. Asadar, serializarea permite "salvarea" valorilor unei variabile.

Daca este serializat un obiect, sunt salvate doar proprietatile acestuia (variabilele membre) si numele clasei din care face parte, nu si metodele deoarece functiile nu reprezinta valori.

Pentru a serializa un obiect, este utilizata functia **serialize()** care returneaza sirul de octeti care contine reprezentarea binara. Pentru a deserializa un obiect, se foloseste functia pereche **unserialize()**. Pentru ca o astfel de operatie sa functioneze corect, este necesara definirea clasei din care face parte obiectul respectiv. Functia returneaza valoarea variabilei serializate.

In exemplul urmator aveti prezentat modul in care poate fi serializat si deserializat un obiect. Sirul de octeti obtinut in urma serializarii va fi scris intr-un fisier si va fi citit din fisierul respectiv pentru efectuarea deserializarii. De obicei serializarea si deserializarea sunt realizate in documente php diferite deoarece aceste operatii nu au aproape nici o utilitate daca sunt folosite in cadrul aceluiasi document.

Primul document in care se realizeaza serializarea trebuie sa contina o secventa asemanatoare cu urmatoarea:

```
<?php

class A {
var $msg = 'Salutare lume';
    function scrie() {
    echo $this -> msg;
    }
}
$a = new A;
$s = serialize($a);
// salvarea sirului intr-un fisier
$fp = fopen ("fisier.txt", "w");
fputs ($fp, $s);
fclose ($fp);

?>
```

Pentru deserializare, al doilea document va contine urmatoarea secventa:

```
class A {
var $msg = 'Salutare lume';
    function scrie() {
    echo $this -> msg;
    }
}
// citirea sirului din fisier
$s = implode('', @file("fisier.txt"));
$a = unserialize($s);
// dupa deserializare obiectul poate fi folosit
// arata unul ..
```

Referintele pot fi utilizate pentru a accesa continutul unei variabile folosind mai multe nume. Spre deosebire de limbajul C, in PHP referintele nu sunt pointeri, ci *alias*-uri intr-o tabela de simboluri. In PHP denumirile variabilelor si continutul acestora nu sunt unul si acelasi lucru. Asadar, este posibil ca acelasi continut sa aiba denumiri diferite.

Utilizarea referintelor

Referintele PHP permit unor variabile cu denumiri diferite sa corespunda unui acelasi continut.

Cu alte cuvinte, instructiunea $a = &$b are ca efect faptul ca $a si $b refera aceeasi variabila. In aceasta situatie $a si $b au acelasi statut. Nu se poate spune ca $a refera $b sau invers.

O alta posibilitate de utilizare a referintelor este transmiterea prin referinta a parametrilor unei functii. Efectul unei astfel de transmisii este crearea unei variabile locale care refera spre acelasi continut ca variabila din contextul apelant.

Sa luam in considerare urmatorul exemplu:

```
function inc(&$var) {
$var++;
}

$a = 5;
inc($a);
```

Initial, valoarea variabilei $a este 5. Dupa apel, variabila locala $var si variabila din contextul apelant $a indica spre acelasi continut. Valoarea pastrata in locatia de memorie respectiva este incrementata (devine 6) prin intermediul instructiunii *$var++*;. Datorita faptului ca cele doua variabile au acelasi continut, valoarea variabilei $a va fi 6 dupa executarea functiei. Un parametru transmis prin referinta poate fi:

- o variabila;
- o instructiune **new**;
- o referinta returnata de o functie.

Daca unei astfel de functii i se transmite ca parametru un alt tip de expresie rezultatul este nedefinit. Asadar, pentru o functie care are un parametru transmis prin referinta, nu se poate folosi o constanta in momentul apelului. De exemplu, pentru functia *inc()* prezentata anterior nu este permis un apel de forma *inc (5)*.

Referinte globale

In momentul declararii unei variabile globale (printr-o instructiune de tipul **global** *$var*) se creeaza de fapt o referinta spre o variabila globala. Cu alte cuvinte, aceasta instructiune este echivalenta cu *$var = &$GLOBALS ["var"];*.

Referinta $this

In cadrul unei metode a unui obiect, **$this** este intotdeauna o referinta spre obiectul care utilizeaza functia (obiectul curent).

Operatori PHP

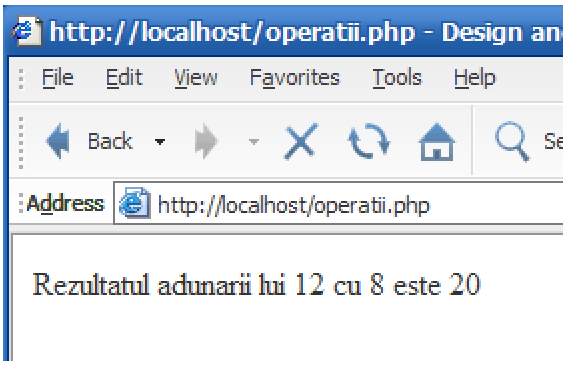
Operatori aritmetici cu variabile

+	Adunare	$a + $b
-	Scadere	$a - $b
*	Inmultire	$a * $b
/	Impartire	$a / $b
%	Modulo	$a % $b

Exemple:

```
<?php
$a = '12';
$b = '8';
$rezultat = $a + $b;
echo 'Rezultatul adunarii lui '.$a.' cu '.$b.' este '.$rezultat.' ';
?>
```

Puneti codul intr-un fisier **operatii.php**, salvati si apoi vizualizati in explorer accesand http://localhost/operatii.php



Operatori de comparatie in PHP

==	Egal	$a == $b
===	Identic	$a === $b
!=	Diferit	$a != $b
<>	Diferit	$a <> $b
<	Mai mic	$a < $b
>	Mai mare	$a > $b
<=	Mai mic sau egal	$a <= $b
>=	Mai mare sau egal	$a >= $b

Operatori logici in PHP

!	NOT	!$b	Returneaza true (adevarat) daca $b este false (fals) si viceversa.
&&	AND	$a && $b	Returneaza true (adevarat) daca atat $a cat si $b sunt true (adevarate) si false (fals) in caz contrar.
||	OR	$a || $b	Returneaza true (adevarat) daca $a, $b sau ambele sunt true (adevarate) si false (fals) in caz contrar.
and	AND	$a and $b	Identic cu && dar are o prioritate mai scazuta.
or	OR	$a or $b	Identic cu || dar are o prioritate mai scazuta.

Operatorul tertiar

conditie?adevarat:fals

```
<?php

$variabila = "oriceon";
echo $variabila == "oriceon" ? "variabila are valoarea oriceon" : "variabila nu are valoarea oriceon";

?>
```

Realizati o pagina cu numele **operator_tertiar.php**, introduceti codul de mai sus si testati in browser pentru a observa rezultatul, apoi modificati valoarea variabilei din oriceon in test, salvati si testati in browser.
Dupa cum vedeti, ruleaza ca instructiunile if si else.
Codul de mai sus se interpreteaza cam asa: daca valoarea variabilei este egala cu valoarea data, in cazul nostru oriceon, atunci printez un text.., daca nu, printez alt text cu un mesaj de eroare…

Luarea deciziilor prin structuri conditionale

Instructiunea IF

Pentru a lua o decizie, in scriptul nostru PHP, putem folosi instructiunea if. Acestei instructiuni trebuie sa ii oferim o conditie pe care sa o foloseasca, iar daca acea conditie este adevarata, va fi executat blocul de cod de dupa ea. Conditiile din instructiunea if trebuie sa fie trecute intre paranteze rotunde ( )

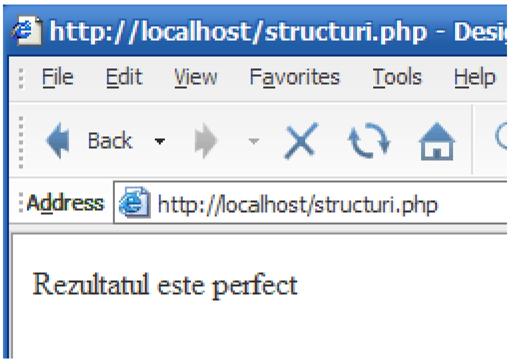
```
<?php

$a = 12;
$b = 8;
$rezultat = $a + $b;

if($rezultat == '20') {
echo 'Rezultatul este perfect';
}

?>
```

Puneti codul intr-un fisier **structuri.php**, salvati si apoi vizualizati in explorer accesand http://localhost/structuri.php



Observati, conditia noastra, si anume aceea ca valoarea rezultata in urma adunarii dintre variabila a (12) si variabila b (8) sa fie egala cu numarul 20, este adevarata si in acest caz, codul de dupa { si respectiv } a fost executat.

Daca valoarea adunarii dintre variabila a si variabila b nu era 20, atunci afisarea in browser era nula.

```
<?php
$a = 155;
$b = 8;
$rezultat = $a + $b;

if($rezultat == '20') {
echo 'Rezultatul este perfect';
}
?>
```

In conditiile noastre, dupa cum vedeti, ne folosim de **operatorii din PHP** pe care i-am scris mai sus

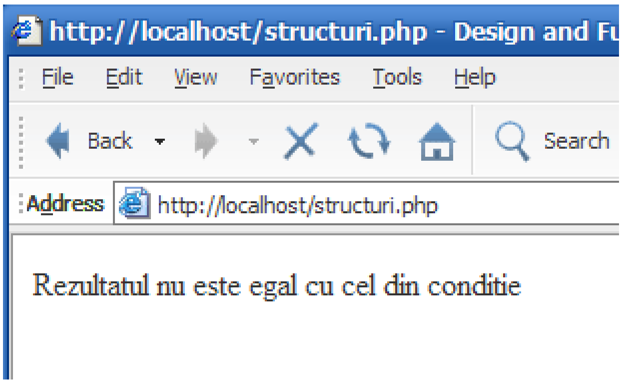
Instructiunea ELSE

De multe ori, pe langa decizia de a executa o actiune, atunci cand conditia este adevarata, doriti sa executati o alta care in caz contrar (in cazul in care conditia nu este adevarata) sa returneze o alta bucata de cod.

```
<?php
$a = 20;
$b = 8;
$rezultat = $a + $b;

if($rezultat == '20') {
echo 'Rezultatul este perfect';
} else {
echo 'Rezultatul nu este egal cu cel din conditie';
}
?>
```

Puneti codul intr-un fisier **structuri.php**, salvati, si apoi vizualizati in explorer accesand http://localhost/structuri.php



Avem aceasta afisare deoarece variabila $a (20) adunata cu variabila $b (8) nu este egal cu 20, deci prin urmare se executa blocul de cod din instructiunea else.

Instructiunea ELSEIF

Aceasta instructiune este (dupa cum vedeti) o combinatie dintre instructiunea if si cea else.
Aceasta poate verifica fiecare conditie pana in momentul in care una dintre contidiile gasite returneaza o valoare adevarata.

```
<?php
$a = 20;
$b = 1;
$rezultat = $a + $b;

if($rezultat == '20') {
echo 'Rezultatul este egal cu 20';
} elseif ($rezultat == '21') {
echo 'Rezultatul este egal cu 21';
} else {
echo 'Rezultatul nu este egal cu cel din conditie';
}
?>
```

Schimbati pe rand valoarea variabilelor si testati in browser.

1) $a = 20;
   $b = 0;

2) $a = 20;
   $b = 1;

3) $a = 20;
   $b = 20;

Pentru varianta 1, 20+0 = 20, se va executa bucata de cod din instructiunea if, daca modificati cu valorile din varianta 2, 20+1 = 21, se va executa bucata de cod din instructiunea elseif, iar daca puneti valorile din varianta 3, va rula instructiunea else.
Toate acestea se interpreteaza si se gandesc cam asa:

Daca prima instructiune este adevarata, afisez ceva, daca nu, verific urmatoarea instructiune si, daca returneaza adevarat, afisez blocul de cod din ea, iar daca nici prima nici a-II-a nu returneaza adevarat, atunci afisez blocul de cod din instructiunea else.

In aceste instructiuni if, else sau elseif ne putem folosi de toti operatorii din PHP pe care i-am spus mai sus.

In exemplul urmator o sa vedem cum calculele se pot complica si vom observa ca intr-o instructiune se pot pune mai multe conditii.

```
<?php
$a = 20;
$b = 1;

$c = 5;
$d = 2;

$rezultat1 = $a + $b;

$rezultat2 = $c - $d;

if(($rezultat1 == '21') || ($rezultat2 != '100')) {
echo 'Rezultatul este ok';
} else {
echo 'Rezultatul nu este egal cu cel din conditie';
}
?>
```

Puneti codul intr-un fisier si vizualizati in browser. Veti observa ca rezultatul o sa fie cel din instructiunea if deoarece returneaza adevarat.

Analizand calculele, vom observa ca rezultat1 este egal cu 21, deoarece adunarea lui $a (20) cu $b (1) ne da 21, iar rezultat2 este egal cu 3, deoarece scaderea lui $c (5) cu $d (2) ne da 3.

Instructiunea noastra if ne rezulta true, deoarece rezultat1 este egal cu 21, prin urmare true, si rezultat2 nu este egal cu 100, deci si aici ne da true.
Amintindu-ne de operatorii din PHP, stim ca || returneaza true (adevarat) daca prima conditie, a II-a conditie sau ambele sunt true (adevarate) si false (fals) in caz contrar.

Schimband instructiunea if in if(($rezultat1 == '21') || ($rezultat2 != '3')), vom observa ca tot bucata de cod din aceasta instructiune, se va executa, chiar daca prima parte este adevarata, iar a-II-a este falsa.

Daca schimbam iar instructiunea if in if(($rezultat1 == '200') || ($rezultat2 != '3')), vom observa ca se va executa urmatoara instructiune, si anume else, deoarece nici prima si nici a-II-a parte nu este valida.

Instructiunea SWITCH

Aceasta instructiune functioneaza asemanator cu cea if, insa permite conditiilor sa aibe mai mult de 2 valori. Intr-o instructiune if, conditia poate fi adevarata sau falsa, insa intr-o instructiune **switch** conditia poate lua orice numar de valori diferite.
Aceasta instructiune trebuie sa contina o instructiune **case** care sa manevreze fiecare valoare pe care o doriti.

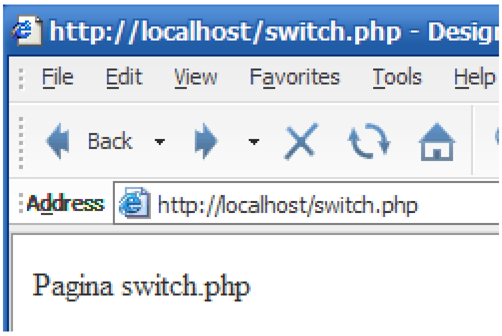
```
<?php

if(!isset($_GET['modul'])) $_GET['modul'] = '';
switch($_GET['modul']) {
      case '':
      echo 'Pagina switch.php';
      break;

      case 'pagina1':
      echo 'Pagina switch.php?modul=pagina1';
      break;

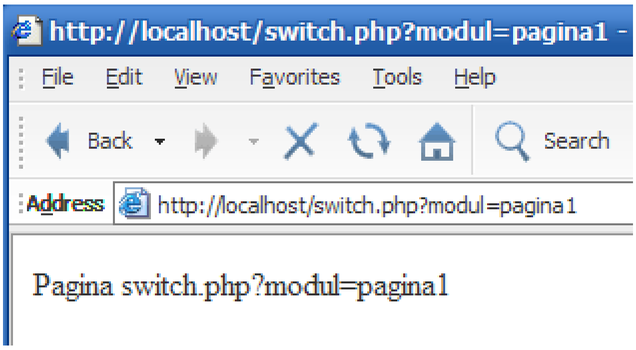
      case 'pagina2':
      echo 'Pagina switch.php?modul=pagina2';
      break;
}
?>
```

Puneti codul intr-un fisier **switch.php**, salvati si apoi vizualizati in browser accesand http://localhost/switch.php



Accesand numai pagina switch.php, se va afisa bucla de cod din interiorul case ‘’:

Acum accesati in browser: http://localhost/switch.php?modul=pagina1 sau ?modul=pagina2



Observati modalitatea de accesare a paginii si rezultatul obtinut.

Bucla WHILE

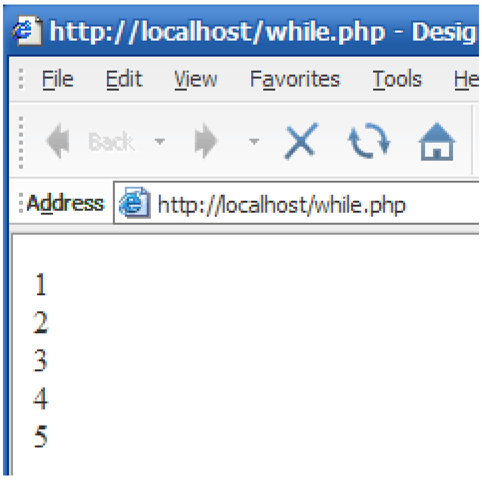
Cel mai simplu tip de bucla PHP este **while.** Asemenea instructiunii if, ea se bazeaza pe o actiune. Diferenta dintre if si while este aceea ca instructiunea if, daca gaseste adevarata conditia, afiseaza o singura data bucata de cod din ea, insa in conditia while, daca rezultatul este adevarat, bucata de cod din ea se va repeta atata timp cat conditia este adevarata.

```
<?php
$numar = 1;

while($numar <= 5)
echo $numar.'<br>';
$numar++;
}

?>
```

Puneti codul intr-un fisier **while.php**, salvati si apoi vizualizati in explorer accesand **http://localhost/while.php**



Structura FOR

O alternativa cu o functionalitate mai ridicata pentru utilizarea buclelor este structura repetitiva **for**.

Sintaxa este foarte asemanatoare cu cea din limbajele *C/C++* si *Java* si anume:

```
for(expresie1; conditie; expresie2) {
//instructiune
}
```

Prima expresie este evaluata o singura data, inainte de inceperea executiei ciclului.
Expresia conditie este testata inaintea fiecarei repetari a buclei. Daca expresia returneaza fals, repetarea se opreste.
Expresia 2 este executata la sfarsitul fiecarei repetari.

Instructiunea se executa la fiecare repetare a buclei.

Oricare dintre cele trei expresii poate lipsi; in cazul in care o expresie lipseste, se considera ca ea are valoarea **true**.

Bucla WHILE si FOR sunt identice din puct de vedere functional insa bucla FOR este putin mai complexa.

```
<?php

for ($variabila = 1; $variabila <= 10; $variabila++) {
    echo $variabila.'<br>';
}

?>
```

Sa mai luam un exemplu de lucru cu bucla for.
Creati o pagina cu numele **for.php**, introduceti codul urmator apoi testati in browser.

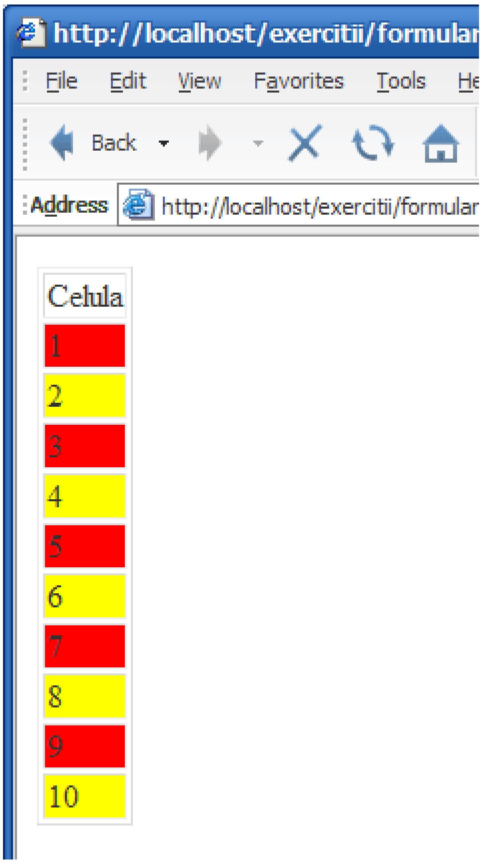
```
<?php

echo "<table border=\"1\">\n<tr><td>Celula</td></tr>\n";

$culoare = "yellow";
for ($variabila = 1; $variabila <= 10; $variabila++) {
    if($culoare == "yellow")  $culoare = "red";
    else $culoare = "yellow";
echo "<tr><td bgcolor=".$culoare.">".$variabila."</td></tr>\n";
}

echo "</table>";

?>
```



Vizualizati si sursa paginii si observati asezarea codului HTML

Structura FOREACH

Aceasta structura poate fi folosita pentru a realiza o repetare printre toate elementele unui vector. Asadar, ea nu poate fi folosita decat impreuna cu vectorii; utilizarea sa asupra unei variabile de alt tip duce la aparitia de erori.

Exista doua sintaxe acceptate pentru aceasta structura si anume:

```
foreach(expresie_vectoriala as $valoare) {
//instructiune
}

foreach(expresie_vectoriala as $cheie => $valoare) {
//instructiune
}
```

Daca se utilizeaza prima varianta, atunci la fiecare iteratie valoarea elementului curent este atribuita variabilei *$valoare,* si apoi se trece la elementul urmator (a carui valoare va fi atribuita variabilei la urmatoarea iteratie).

Executia ciclului se incheie in momentul in care nu mai exista alte elemente in vector. Singura diferenta care apare in cazul utilizarii celei de-a doua variante este faptul ca la fiecare iteratie valoarea cheii elementului curent este atribuita variabilei *$cheie*.

In continuare este un exemplu de folosire a celor doua sintaxe ale structurii **foreach**.

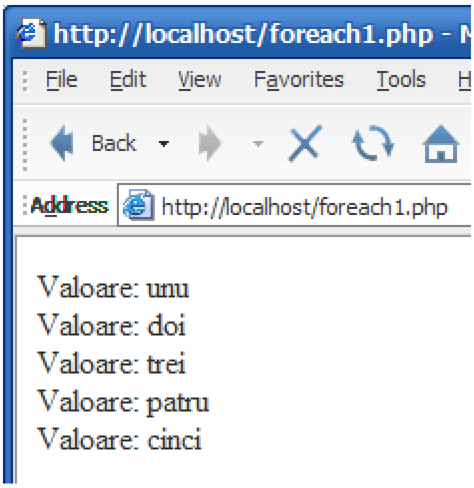
```
<?php

$sir = array("unu", "doi", "trei", "patru", "cinci");

foreach($sir as $valoare) {
echo "Valoare: ".$valoare." <br>\n";
}

?>
```

Realizati o pagina cu numele **foreach1.php**, introduceti codul de mai sus si testati in browser.



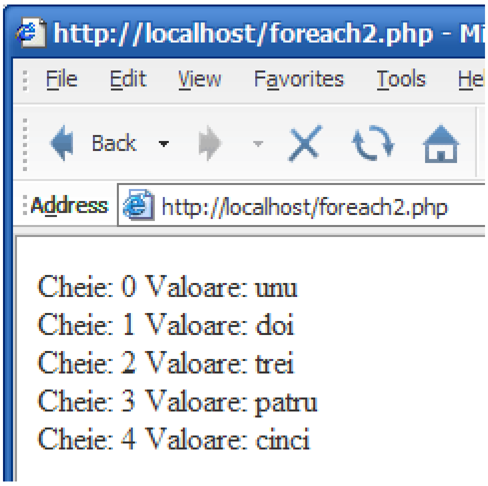
Acum, sa realizam cel de-al II-lea exemplu; creati o pagina cu numele **foreach2.php**, introduceti codul de mai jos apoi testati in brwoser.

```php
<?php

$sir = array("unu", "doi", "trei", "patru", "cinci");

foreach($sir as $cheie => $valoare) {
echo "Cheie: ".$cheie." Valoare: ".$valoare." <br>\n";
}

?>
```



Observati diferenta dintre primul exemplu de foreach si cel de-al II-lea.

Instructiunea BREACK

Aceasta instructiune poate fi folosita pentru a intrerupe fortat executia unui ciclu sau a secventei de instructiuni corespunzatoare unei structuri **switch**.

Instructiunea poate fi urmata de un argument care indica numarul de structuri imbricate a caror executie se incheie. Valoarea implicita este 1, deci se intrerupe executia unei singure structuri. Urmatoarea secventa de cod PHP realizeaza parcurgerea elementelor unui vector de numere intregi, pana in momentul in care se intalneste un numar negativ.

```
foreach ($a as $v)
if($v < 0)
break;
```

Mai departe, aveti cazul in care este intrerupta executia mai multor cicluri; vom considera ca parcurgem elementele unei matrice patratice cu n elemente si n coloane pana in momentul in care intalnim o valoare nula.

```
for($i = 0; $i < $n; $i++)
for($j = 0; $j < $n; $j++)
if(!$a[$i][$j])
break 2;
```

Instructiunea **break** poate fi utilizata pentru intreruperea executiei secventelor de instructiuni corespunzatoare structurilor **for**, **foreach**, **while**, **do - while** si **switch**.

Instructiunea CONTINUE

Aceasta instructiune este folosita pentru a intrerupe executia secventei de instructiuni din interiorul unui ciclu si trecerea la urmatoarea iteratie.

In cazul instructiunii **for**, inainte de urmatoarea iteratie se evalueaza (executa) expresia de incrementare (expresia #3 din sintaxa generala). La fel ca si in cazul instructiunii **break**, poate aparea un argument care indica numarul structurilor imbricate asupra carora are efect.

Exemplul urmator realizeaza afisarea elementelor unui sir de numere intregi care sunt mai mari decat 1000.

```
foreach($a as $v) {
if($v <= 1000)
continue;
echo $v;
}
```

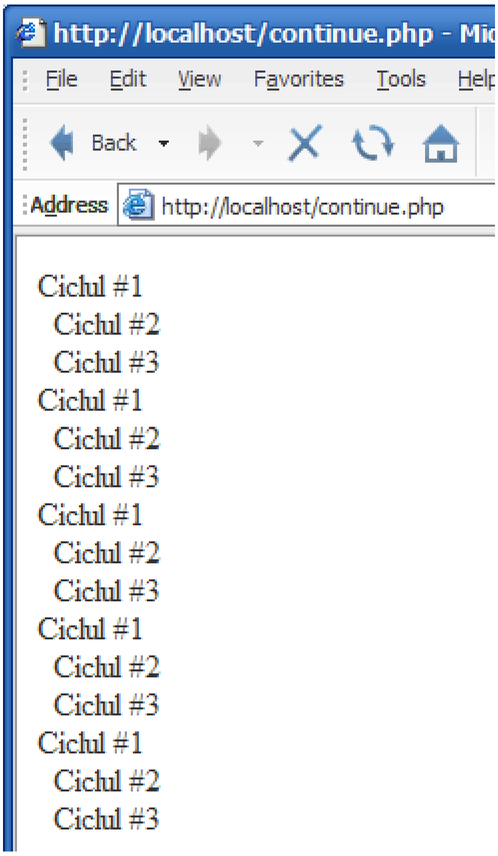
Urmatorul exemplu ilustreaza efectul folosirii argumentelor pentru instructiunea **continue**.

```
<?php

$i = 0;
while($i++ < 5) {
echo "Ciclul #1 <br>\n";
    while(1) {
    echo "  Ciclul #2 <br>\n";
        while (1) {
        echo "  Ciclul #3<br>\n";
        continue 3;
        }
    echo "Acest mesaj nu va fi afisat niciodata.<br>\n";
    }
echo "Nici acest mesaj nu va fi afisat niciodata.<br>\n";
}

?>
```

Realizati o pagina cu numele **continue.php**, introduceti codul de mai sus apoi testati in browser.



Alte structuri PHP

Exista mai multe alte structuri PHP care pot fi utilizate in anumite scopuri.

Vom aminti acum cateva dintre ele:

Structurile **include**, **require**, **include_once** si **require_once** pot fi utilizate pentru a "insera" anumite instructiuni care sunt pastrate intr-un alt fisier (document). Interpretorul PHP considera ca secventa din fisierul inserat se afla in fisierul din care s-a "comandat" inserarea in pozitia in care apare structura de inserare.

O alta structura este **declare** care permite crearea unor directive in executie.

Functiile PHP trebuie sa utilizeze instructiunea **return** pentru a furniza un rezultat.

Prelucrarea datelor printr-un formular

Inainte de a vorbi despre prelucrarea datelor printr-un formular, trebuie in primul rand sa cunoasteti sintaxa HTML pentru crearea unui formular.

Un formular este delimitat de elementul FORM care contine alte cateva elemente numite “controale”, care au o varietate de metode de a aduna informatii. Fiecare element din formular are un nume cat si o valoare, astfel incat datele transferate pentru procesare sa fie sub forma unor perechi nume/valoare.

Elementul FORM

<form [action=url] [method=get/post] [enctype=MIMEType] [onsubmit=script] [onreset=script] [accept-charset=set_caractere] [core] [international] [events]>

Elementele formularului

</form>

Elementul INPUT

<input [type=text|password|checkbox|radio|submit|image|reset|button|hidden|file] [name=nume] [value=valoare] [checked] [disabled] [readonly] [size=latime] [maxlength=cuvinte_maxime] [src=url] [alt=altText] [usemap=url] [align=left|center|right|justify] [tabindex=numar] [accesskey=keyCombo] [onfocus=script] [onblur=script] [onselect=script] [onchange=script] [accept=set_caractere] [core] [international] [events]>

Acest element input este cel mai important in utilizarea formularelor.

Explicarea valorilor Type ale elementului INPUT

button   Butoane personale   Exemplu: <input type=”submit” name=”Buton” value=”Buton”>



checkbox   Casete de validare   Exemplu: <input type=”checkbox” name=”nume” value=”valoare”>



file   Fisiere incluse   Exemplu: <input type=”file” name=”nume” value=”valoare”>



hidden   Elemente ascunse   Exemplu: <input type=”hidden” name=”nume” value=”valoare”>

image   Imagini   Exemplu: <input type=”image” name=”Buton” src=”poza_buton.gif”>



password   Casete de introducere a parolei   <input type=”password” name=”nume” value=”valoare”>



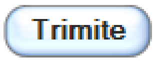
radio   Butoane radio   Exemplu: <input type=”radio” name=”nume” value=”valoare”>



reset Buton reset Exemplu: <input type=”reset” name=”Reseteaza” value=”Reseteaza”>



submit Buton trimite Exemplu: <input type=”button” name=”Trimite” value=”Trimite”>



text Caseta text Exemplu: <input type=”text” name=”nume” value=”valoare”>



Elementul SELECT

Acest element este folosit pentru crearea unei liste de optiuni, fie ca un meniu care se desfasoara, fie ca o caseta cu lista. Fiecare din optiunile din lista reprezinta un element OPTION.

<select [name=nume] [size=latime] [multiple] [disabled] [tabindex=numar] [onfocus=script] [onblur=script] [onchange=script] [core] [international] [events]>

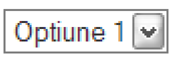
Elementele din select

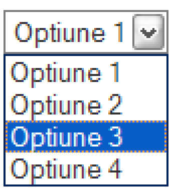
</select>

<option [selected] [disabled] [value=valoare] [core] [international] [events]>Nume</option>

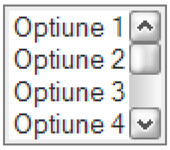
Exemplu select:

<select name=”test”>
<option value=”optiune 1”>Optiune 1</option>
<option value=”optiune 2”>Optiune 2</option>
<option value=”optiune 3”>Optiune 3</option>
<option value=”optiune 4”>Optiune 4</option>
</select>





```
<select name="test" multiple size="3">
<option value="optiune 1">Optiune 1</option>
<option value="optiune 2">Optiune 2</option>
<option value="optiune 3">Optiune 3</option>
<option value="optiune 4">Optiune 4</option>
<option value="optiune 5">Optiune 5</option>
<option value="optiune 6">Optiune 6</option>
</select>
```



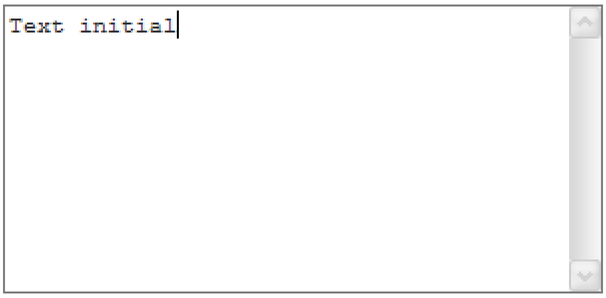
Elementul TEXTAREA

Acest element este asemanator cu cel text numai ca aici se poate tasta intr-o sectiune mult mai mare decat in cazul text.

```
<textarea [name=nume] [rows=nr_randuri] [cols=nr_coloane] [disabled] [readonly] [tabindex=numar]
[onfocus=script] [onblur=script] [onselect=script] [onchange=script] [core] [international] [events]</textarea>
```

Exemplu textarea:

```
<textarea name="nume" cols="40" rows="10">Text initial</textarea>
```



Exemplu de formular:

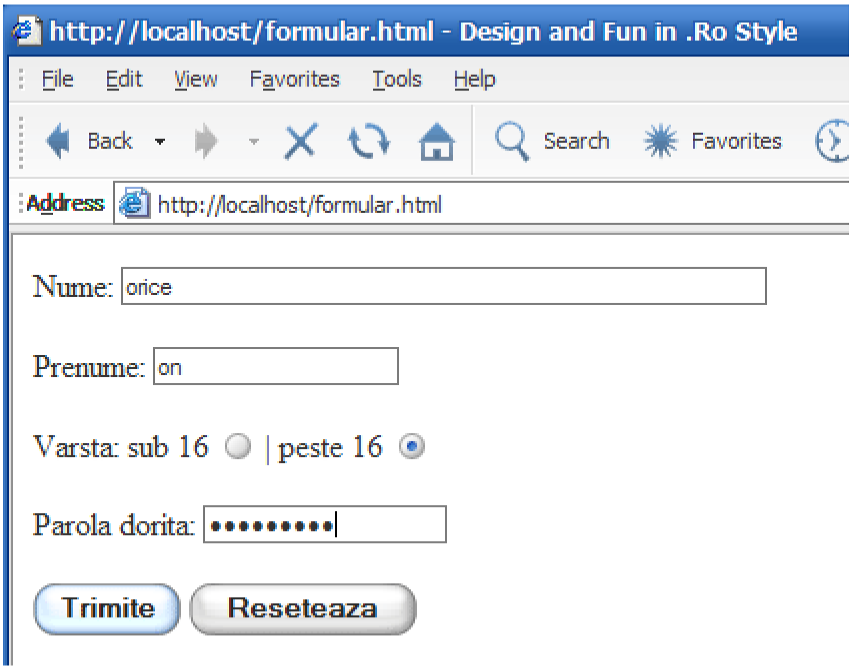
```
<form action="formular.php" method="post">

Nume: <input type="text" name="nume" value="" size=60 maxLength=15> <br><br>
Prenume: <input type="text" name="prenume" value=""> <br><br>
Varsta:  sub 16 <input type="radio" name="varsta" value="sub16"> | peste 16
                <input type="radio" name="varsta" value="peste16"> <br><br>
Parola dorita: <input type="password" name="parola" value=""> <br><br>

<input type="submit" name="Trimite" value="Trimite">
<input type="reset" name="Reseteaza" value="Reseteaza">

</form>
```

Puneti codul intr-un fisier **formular.html**, salvati si apoi vizualizati in explorer accesand http://localhost/formular.html



Prelucrarea datelor din formular

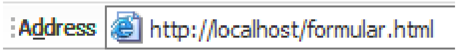
Am vorbit pana acum despre formulare, cum sa le cream si ce elemente au.
Hai sa ne aducem aminte ca in interiorul etichetei <form> avem pus elementul action si method (care poate fi POST sau GET). In elementul action se pune calea catre scriptul PHP care prelucreaza datele iar in method se pune metoda prin care se vor prelucra datele atunci cand butonul “Trimite” este apasat.

POST

Aceasta metoda face ca datele trimise prin formular sa nu fie vizibile utilizatorului, sa fie trimise in spatele paginii web.

GET

Prin aceasta metoda, datele trimise prin formular sunt vizibile in URL (URL este adresa ce este afisata in browser)



Acum ca am creat formularul, avem pagina formular.html, haideti sa cream si pagina prin care aratam datele trimise prin formular.

Creati o pagina formular.php si introduceti codul urmator:

```
<?php

echo $_POST['nume'];

?>
```

Salvati si testati in browser accesand http://localhost/formular.html, completati casuta “Nume”, si apoi apasati butonul “Trimite”.

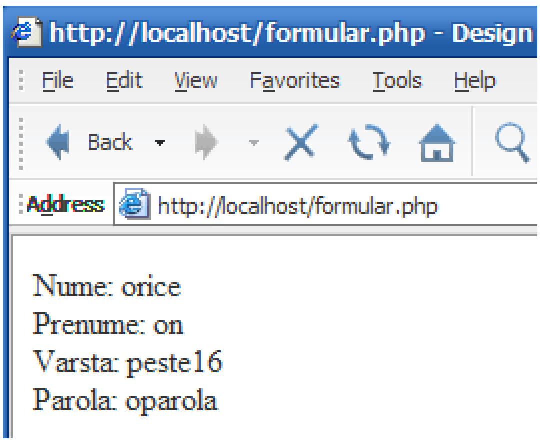
Observati ca in pagina formular.php ne este afisat numele introdus in casuta “Nume” din pagina anterioara, si anume formular.html.

Adaugam in continuare variabilele $_POST corespunzatoare formularului nostru, si apoi testam din nou.

```
<?php

echo 'Nume:       '.$_POST['nume'].'       <br>
      Prenume:    '.$_POST['prenume'].'    <br>
      Varsta:     '.$_POST['varsta'].'     <br>
      Parola:     '.$_POST['parola'].'     <br>';

?>
```



Dupa cum vedeti, ne folosim de valoarea POST, iar datele nu sunt afisate decat daca le definim noi $_POST[‘nume’] in pagina formular.php.

Poate ca va intrebati de unde pun eu numele in interiorul variabilei $_POST.
Raspunsul consta in numele din interiorul elementelor din formular.

De exemplu, in formularul nostru avem input-ul de la nume asa:
<input type="text" name="nume" value="" size=60 maxLength=15>

Observati ca in campul name valoarea acestuia este nume. Acesta se plaseaza in variabila $_POST, in cazul nostru $_POST[‘nume’]

In input-ul pentru prenume avem <input type="text" name="prenume" value=""> iar acesta se plaseaza in

$_POST asa: $_POST[‘prenume’] . . . si tot asa.

Observati campul value care nu are nimic definit.
Nu este nici o greseala, veti invata mai tarziu de ce am lasat gol acel camp.

Am cam terminat cu metoda **POST**, acum haideti sa lucram si sa ne familiarizam si cu metoda **GET**.

In acelasi formular din pagina formular.html, numai ca in eticheta <form>, in loc de **method=”post”,** vom pune **method=”get”**.
Apoi in pagina formular.php, in loc de variabilele **$_POST,** vom pune variabile **$_GET**.

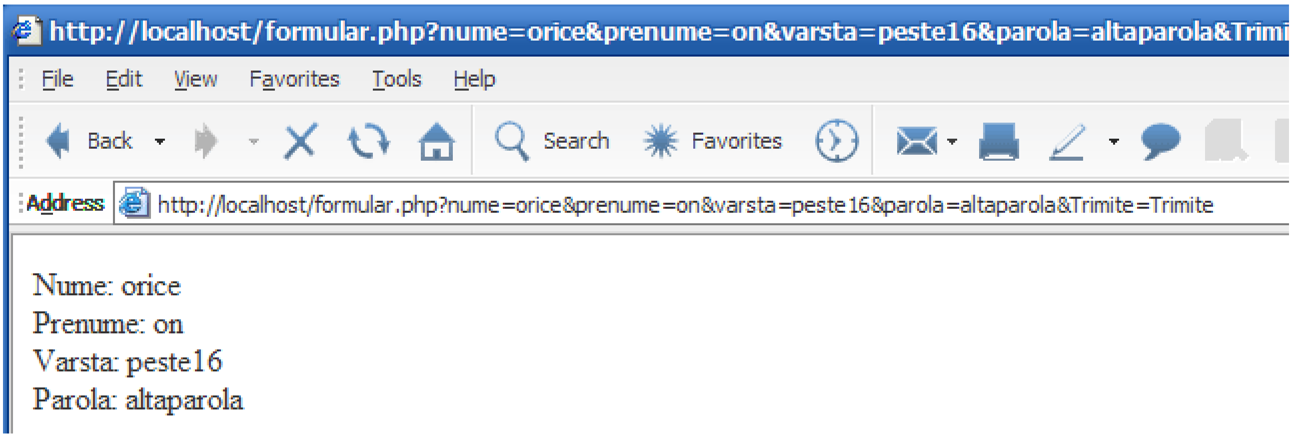
In caz ca nu ati inteles, deschideti pagina formular.html, modificati linia <form action="formular.php" method="post"> in <form action="formular.php" method="get">

Apoi deschideti pagina formular.php si modificati in:

```
<?php

echo 'Nume:      '.$_GET['nume'].'      <br>
      Prenume:  '.$_GET['prenume'].'   <br>
      Varsta:   '.$_GET['varsta'].'    <br>
      Parola:   '.$_GET['parola'].'    <br>';

?>
```



Observati ca rezultatul este acelasi in pagina web, numai ca informatiile formularului sunt postate si in adresa din browser (url) sub forma:
http://localhost/formular.php?nume=orice&prenume=on&varsta=peste16&parola=altaparola&Trimite=Trimite

De recomandat este sa folositi metoda **POST,** pentru ca este mult mai sigura.

Verificarea datelor trimise prin formular

Verificarea continutului trimis prin formular se face relativ foarte usor.

```
<?php

if(($_POST['nume'] == "") || (is_numeric($_POST['nume'])) ||
(strlen($_POST['nume']) < 5)) {

echo 'Campul nume nu a fost completat corect. <br>
      <a href="formular.html">Apasa aici</a> pentru a te intoarce la formular.';
} else {
echo 'Nume:     '.$_POST['nume'].'     <br>
      Prenume: '.$_POST['prenume'].' <br>
      Varsta:  '.$_POST['varsta'].'  <br>
      Parola:  '.$_POST['parola'].'  <br>';
}

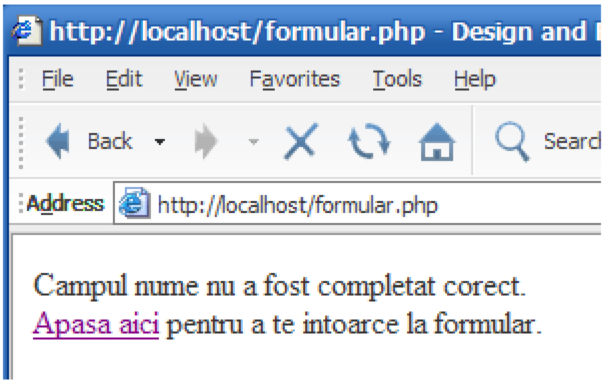
?>
```

Dupa cum vedeti, ne folosim de constructia IF si ELSE, iar in interiorul instructiunii if avem conditiile (momentan numai pentru casuta nume) si operatorii PHP invatati la inceputul tutorialului.

Conditia noastra if spune cam asa: daca variabila cu numele post este goala, ori daca este numerica, ori daca numarul de caractere este mai mic de 5, rezulta bucla de cod din instructiunea if.
Daca toate acestea sunt indeplinite, atunci execut bucla de cod din instructiunea else.

||	OR	$a || $b	Returneaza true (adevarat) daca $a, $b sau ambele sunt true (adevarate) si false (fals) in caz contrar.

Modificati valoarea lui method din pagina formular.html in post si apoi in pagina formular.php introduceti scriptul de mai sus pentru a testa regulile pentru campul “Nume”.



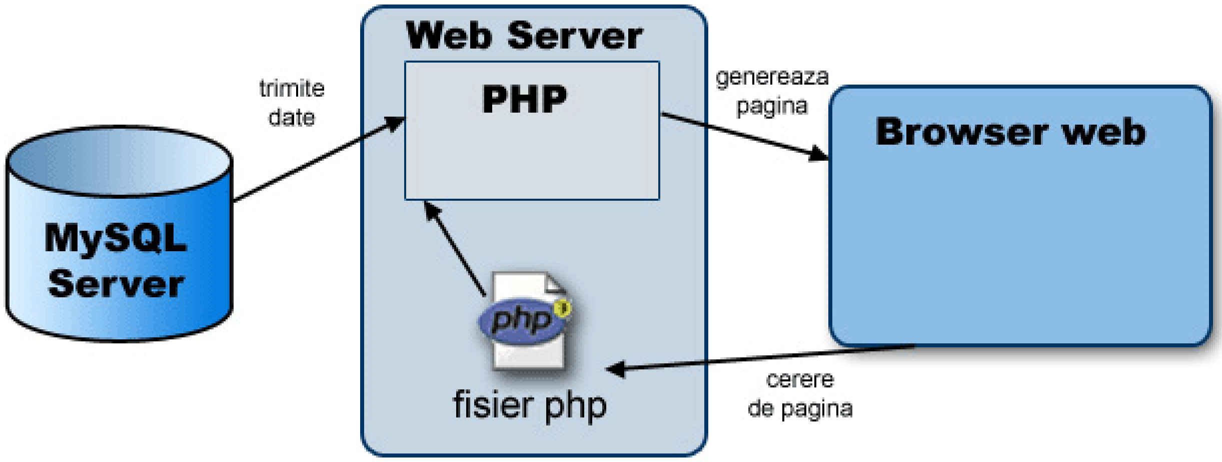
Daca conditiile din if nu sunt nu rezulta true, atunci va rezulta acest mesaj.

In conditia if, puteti continua sirul de conditii . . de exemplu:
if(($_POST['nume'] == "") || (is_numeric($_POST['nume'])) || (strlen($_POST['nume']) < 5) || ($_POST['prenume'] == "") || ($_POST['parola'] == "")) etc etc

(is_numeric($_POST['nume'])) – aceasta parte ne spune daca valoarea campului nume este numerica.

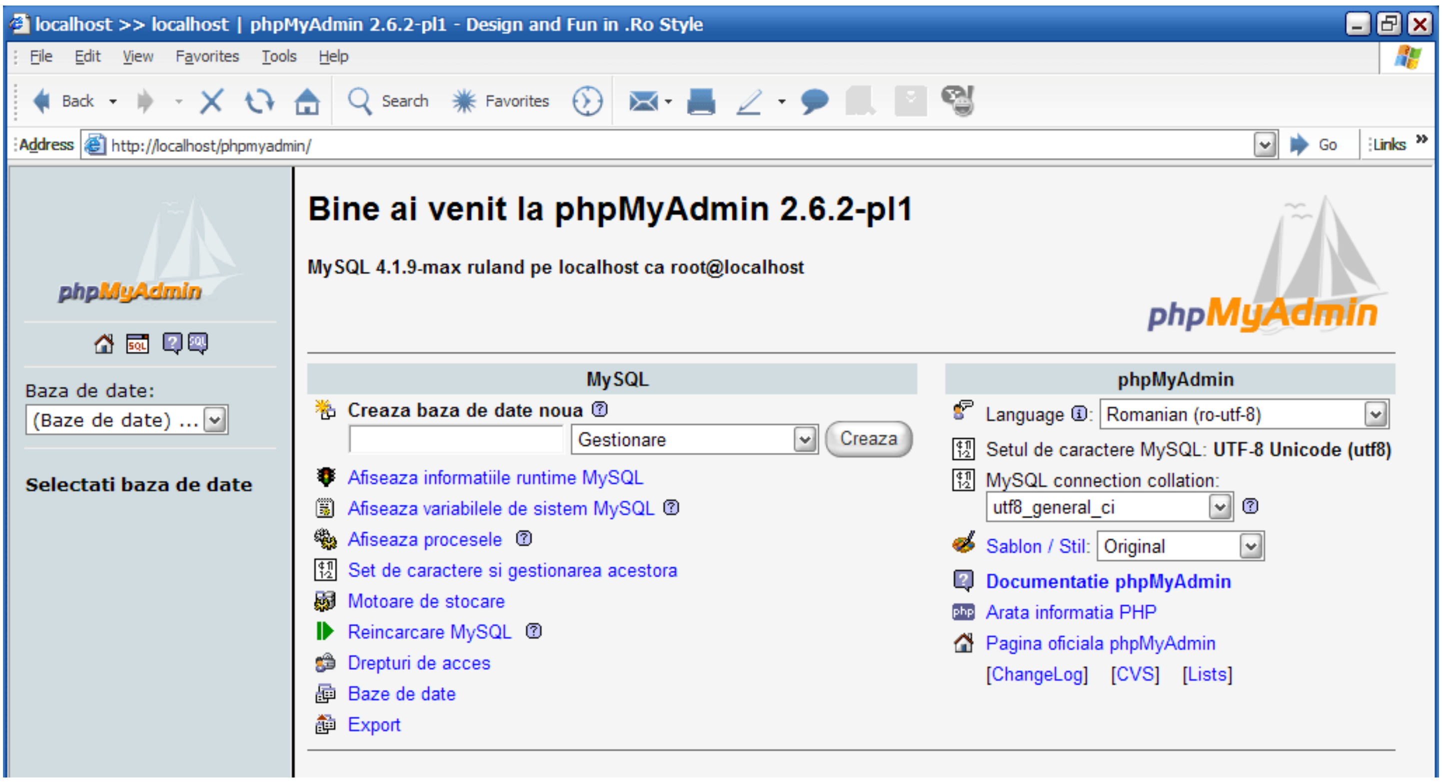
Putem folosi si varianta in care dorim sa stim daca nu este numerica, si anume:
(!is_numeric($_POST['nume'])) observati ca am adaugat un ! pentru a nega (acest lucru si alte combinatii le puteti folosi pe cele din **operatorii PHP** din inceputul tutorialului).
Pana acum am invatat cum sa postam valorile din formular de pe o pagina pe alta fara sa le stocam undeva. Partea cu stocarea o sa o invatam putin mai incolo cand o sa studiem lucrul cu baza de date MySQL.

Lucrul cu baza de date MySQL



Acesta este modul in care o baza de date lucreaza cu un server.

La inceputul tutorialului v-am spus ceva de phpMyAdmin. Acesta este un pachet de scripturi php care va ajuta sa gestionati baza de date folosind o interfata web.
In poza de mai jos va este aratat cum arata acest pachet de scripturi php numit phpMyAdmin.



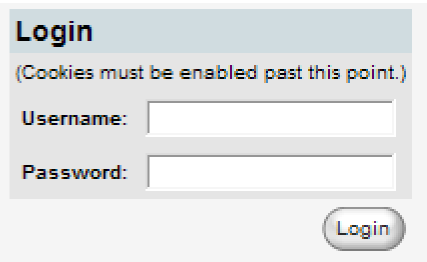
Dupa ce ati descarcat de pe internet ultima versiune de phpMyAdmin, fiti siguri ca ati dezarhivat si ati pus folderul in directorul www.
Intrati apoi in folderul phpMyAdmin si deschideti fisierul config.inc.php. Cautati si editati liniile 84, 85 si 86 care arata cam asa:

```
$cfg['Servers'][$i]['auth_type']     = 'config';     // Authentication method (config, http or cookie based)?
$cfg['Servers'][$i]['user']          = 'root';       // MySQL user
$cfg['Servers'][$i]['password']      = '';           // MySQL password (only needed
```

Modificati modul de autentificare si alegeti in loc de config, cookie astfel incat linia sa fie:
`$cfg['Servers'][$i]['auth_type']     = 'cookie';`
apoi setati o parola secreta pentru modul de autentificare. Cautati linia `$cfg['blowfish_secret'] = '';` si setati in interiorul ei un cuvant, de exemplu `$cfg['blowfish_secret'] = 'baubau';`

Dupa ce ati facut aceste modificari, este recomandat ca in locul user-ului root, sa creati un altul cu mai putine drepturi. Intrati in phpMyAdmin ca root si creati un nou utilizator apoi setati-l in config.inc.php.

Accesati apoi http://localhost/phpmyadmin si observati ca va cere un user si o parola pentru a va autentifica:



Introduceti username root iar ca parola, introduceti cea setata de dumneavoastra.

Daca ati instalat EasyPHP, acest pachet nu va seteaza o parola de inceput (default) pentru baza de date, acest lucru il faceti dumneavoastra.

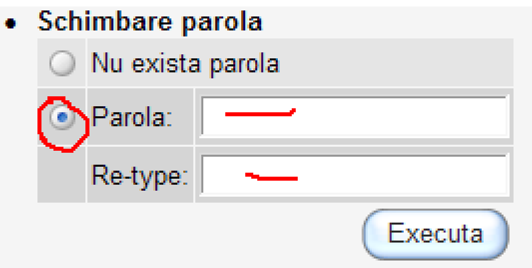
Fara sa faceti vreo modificare in fisierul acesta de configurare, accesati http://localhost/phpmyadmin dupa ce ati pus folderul phpMyAdmin in www.

Dupa ce s-a incarcat pagina, apasati pe legatura “Drepturi de acces” (daca interfata este in limba romana) daca este in engleza, apasati pe “Privileges”. Acest buton se afla in mijlocul paginii.

Va va incarca pagina cu drepturile de acces asupra MySQL-ului, apoi apasati pe butonul ce l-am incercuit eu in poza de mai jos.

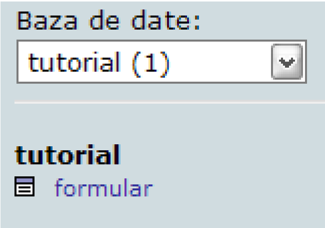
	Utilizator	Gazda (Host)	Parola	Privilegii globale	Grant	
☐	root	localhost	Da	ALL PRIVILEGES	Da	

Apoi in dupa ce s-a incarcat urmatoarea pagina, gasiti unde apare **”Schimbare parola”** si bifati ”Parola”, apoi tastati parola dorita de dumneavoastra si in prima, si in a-II-a casuta, dupa care apasati pe butonul „Executa”.



De acum aveti setata o parola la baza de date MySQL, urmatorul pas este sa ne intoarcem la fisierul config.inc.php si sa ii setam parola in campul $cfg['Servers'][$i]['password'] = ‘’; si anume $cfg['Servers'][$i]['password'] = ‘parolamysql’;

O baza de date este coloana vertebrala a unui site dinamic.
Ea este alcatuita din tabele care, la randul lor, sunt formate din inregistrari dispuse in campuri.



	Camp	Tip	Gestionare	Proprietati	Nul	Setare de baza	Extra	Actiune
☐	id	int(11)			Nu		auto_increment	
☐	nume	char(30)	latin1_swedish_ci		Nu			
☐	prenume	char(30)	latin1_swedish_ci		Nu			
☐	varsta	char(5)	latin1_swedish_ci		Nu			

Dupa cum vedeti, in prima poza apare baza de date cu numele “tutorial” cu tabela “formular”.
In a-II-a poza, apar inregistrarile (coloanele) din cadrul tabelei formular.

Un rand din baza de date se alcatuieste din:

1) Un **nume**, dupa cum vedeti in coloana **camp**. Acest nume nu poate contine spatiu.
2) O valoare **tip** care difera coloana la coloana.
Cele mai folosite tipuri sunt:

Tipuri numerice:
INT – Stocare octeti 4
BIGINT – 64 biti

Tipuri de sir:
CHAR – Interval 1-255 caractere
VARCHAR – Interval 1-255 caractere

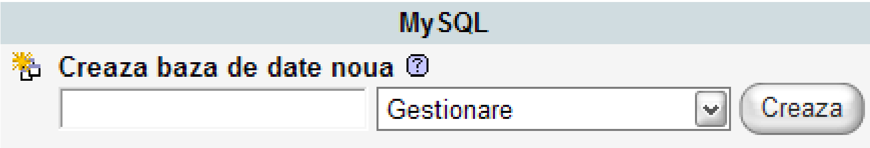
Tipuri de text:
TEXT – Lungime maxima de caractere 65.535
LONGTEXT – Lungime maxima de caractere 4.294.967.295

Binenteles, se pot folosi si alte tipuri pentru datele introduse in baza de date, insa o sa lucram numai cu acestea.

Exemple de folosire phpMyAdmin

1) Crearea unei baze de date

Accesati http://localhost/phpmyadmin/ si apoi o sa observati in mijloc, campul “Creaza baza de date noua”.

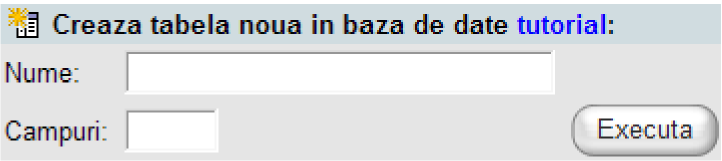


Scrieti numele bazei de date pe care doriti sa o creati,dupa care apasati butonul “Creaza”.

Sa luam ca exemplu, baza de date cu numele tutorial

Dupa ce ati creat baza de date cu numele tutorial, veti observa ca in partea stanga apare numele acesteia, iar in mijloc este scris textul “Nu s-a gasit nici un tabel in baza de date.”.

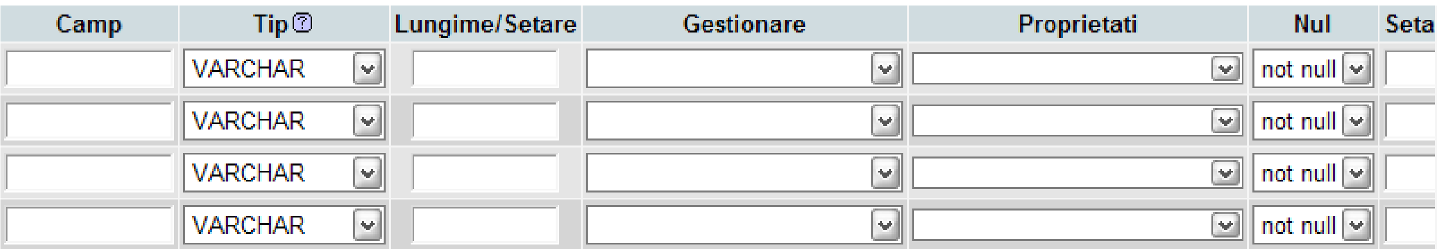
Pentru a adauga un tabel in baza noastra de date, trebuie sa complectam campul “Creaza tabela noua in baza de date tutorial:”.



In casuta campul “Nume” introduceti numele tabelei ce va apare in baza de date tutorial, iar in casuta “Campuri” introduceti numarul de randuri ce le va avea aceasta tabela.

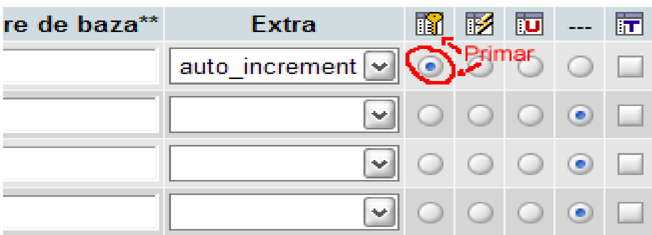
Sa luam ca exemplu: Nume: formular
Campuri: 4

Apasam butonul “Executa”



Intorcandu-ne putin la cunostintele pe care le-am acumulat cu o pagina mai sus, vom observa ca apar campurile: “Camp”, “Tip”, “Lungime”. . .

O prima adaugare in orice baza de date, este bine sa fie un camp cu numele **id**, iar ca tip sa fie **INT**, si o setare speciala pentru aceasta prima linie din tabela noastra, este alegerea valorii **auto_increment** din categoria “Extra”, si apoi trebuie sa bifati optiunea **primar**



Dupa care adaugati pe rand in coloana “Camp” urmatoarele: nume , prenume , varsta.
Ca Tip pentru aceste 3 intrari, alegeti CHAR, apoi in coloana “Lungime/Setare” adaugati o valoare numerica, adica numarul de caractere maxime care sa poata intra in acel rand.

Exemplu: Pentru coloana camp cu valoarea “nume” o sa avem ca tip CHAR si lungime 30. Asta inseamna ca putem introduce un text mai mic sau egal cu 30 caractere.

Camp	Tip	Lungime/Setare	Gestionare	Proprietati	Nul	Seta
id	INT				not null	
nume	CHAR	30			not null	
prenume	CHAR	30			not null	
varsta	CHAR	6			not null	

re de baza**	Extra				---	
	auto_increment	(•) Primar	( )	( )	( )	☐
		( )	( )	( )	(•)	☐
		( )	( )	( )	(•)	☐
		( )	( )	( )	(•)	☐

Dupa ce ati introdus datele precum am spus mai sus si am aratat in poze, apasati butonul “Salveaza”.

In urmatoarele 2 poze vedeti ca tabelul a fost creat, vi se arata comanda **SQL**, dupa care vi se listeaza aceasta tabela.

Tabel formular a fost creat.

Comansa SQL:

```
CREATE TABLE `formular` (
  `id` INT NOT NULL AUTO_INCREMENT ,
  `nume` CHAR( 30 ) NOT NULL ,
  `prenume` CHAR( 30 ) NOT NULL ,
  `varsta` CHAR( 6 ) NOT NULL ,
  PRIMARY KEY ( `id` )
) TYPE = MYISAM ;
```

[Editare] [Creaza cod PHP]

	Camp	Tip	Gestionare	Proprietati	Nul	Setare de baza	Extra	Actiune
☐	id	int(11)			Nu		auto_increment	
☐	nume	char(30)	latin1_swedish_ci		Nu			
☐	prenume	char(30)	latin1_swedish_ci		Nu			
☐	varsta	char(6)	latin1_swedish_ci		Nu			

Dupa ce ati creat baza de date cu numele tutorial si tabela cu numele formular, doriti sa adaugati informatii in aceasta tabela.

Sus in pagina aveti un meniu:

Structura | Navigare | SQL | Cauta | Inserare | Export | Operatii | Goleste | Arunca

Apasati pe butonul “Inserare”

Camp	Tip	Functie	Nul	Valoare
id	int(11)			
nume	char(30)			
prenume	char(30)			
varsta	char(6)			

Completati campul “Valoare” cu datele pe care doriti sa le introduceti in formular pentru fiecare in valorile casutei “Camp”.

Si anume:

Camp	Tip	Functie	Nul	Valoare
id	int(11)			
nume	char(30)			orice
prenume	char(30)			on
varsta	char(6)			20 Ani

Nu complectati casuta id de dip INT deoarece aceasta este folosita pentru a adauga un numar UNIC pentru fiecare intrare din baza de date.

De exemplu, daca adaugati 2 intrari, numarul primei intrari va fi, 1 iar urmatoarei intrari va fi 2. Daca stergeti intrarea cu numarul 2 si adaugati o noua intrare, id-ul acesteia va fi 3 nu 2, deoarece valoarea campului id de tip INT nu se updateaza (in sensul ca se rearanjeaza numarul din dreptul fiecarei intrari) in momentul in care se sterge o intrare din tabela.

Dupa ce ati adaugat informatii in casuta “Valoare”, apasati butonul “Executa”.

Randuri inserate: 1
ID rand inserat: 1

Comansa SQL:

```
INSERT INTO `formular` ( `id` , `nume` , `prenume` , `varsta` )
VALUES (
  '', 'orice', 'on', '20 Ani'
);
```

[Editare] [Creaza cod PHP]

La fel ca si in atunci cand am adaugat o tabela, ni se va arata sintaxa **SQL**

Pentru a vizualiza datele adaugate in tabela formular, apasati pe butonul “Navigare” din partea de sus a pagini.

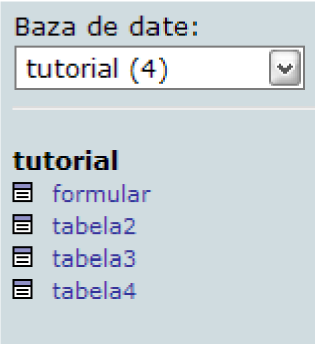
←T→	id	nume	prenume	varsta
	1	orice	on	20 Ani

In aceasta instanta vedeti ca puteti edita, sterge aceasta intrare.

Mai adaugati o intrare in acest tabel efectuand acelasi procedeu ca cel de mai sus si apasati din nou butonul navigare.
Apasati pe textul “nume” sau “prenume” sau “varsta” si observati ca listarea intrarilor se modifica in ordinea adaugarii lor in tabela. Sus observati sintaxa **SQL**

Listarea poate fi **ASC** (ascendenta), adica afisarea intrarilor pornind cu ultima intrare si sfarsind cu prima intrare, sau **DESC** (descendenta), adica afisarea intrarilor pornind cu prima intrare si sfarsind cu ultima.

Nota: Puteti adauga in continuare si alte tabele si coloane atatea cate aveti nevoie.



Pana acum am invatat cum sa lucram cu baza de date folosind phpMyAdmin. Inainte de a trece mai departe va rog efectuati cateva teste adaugand/stergand baze de date, adaugand tabele, modificand…

phpMyAdmin nu este foarte greu atata timp cat stiti ROMANA si stiti sa cititi.

Conectarea la MySQL folosind PHP

In acest capitol vom invata cum sa ne conectam la o baza de date, cum sa citim informatii din ea, cum sa le stergem/modificam sau cum sa adaugam noi informatii cu ajutorul scripturilor PHP.

Creati un folder in directorul radacina a serverului dumneavoastra apache (www) cu numele **tutorial**.
Creati un fisier cu numele config.php in care puneti urmatorul cod:

```php
<?php

// Informatii baza de date

$AdresaBazaDate = "localhost";
$UtilizatorBazaDate = "root";
$ParolaBazaDate = "parola_mysql";
$NumeBazaDate = "tutorial";

$conexiune = mysql_connect($AdresaBazaDate,$UtilizatorBazaDate,$ParolaBazaDate)
or die("Nu ma pot conecta la MySQL!");
mysql_select_db($NumeBazaDate,$conexiune)
or die("Nu gasesc baza de date!");

?>
```

Acesta este fisierul de configurare cu care vom realiza conexiunea la baza noastra de date.
Modificati valoarea variabilei $ParolaBazaDate cu parola pe care ati setat-o dumneavoastra bazei de date. (vezi pagina 30).

Variabila `$AdresaBazaDate` este definita cu valoarea localhost deoarece aceasta este adresa serverului. (Adica, serverul Apache+PHP este instalat pe acelasi calculator ca si pachetul MySQL)

Variabila `$UtilizatorBazaDate` este definita cu valoarea root, acesta fiind utilizatorul cu toate drepturile de acces asupra bazei de date, administratorul.

Variabila `$NumeBazaDate` este definita cu valoarea tutorial, aceasta fiind numele bazei de date asupra careia lucram.

Salvati, si inchideti fisierul.

SELECT

Creati un nou fisier cu numele **extragere_date.php** si introduceti in el urmatorul cod:

```php
<?php
require_once('config.php');

// Selectare dare din baza de date

$cerereSQL = 'SELECT * FROM `formular`';
$rezultat = mysql_query($cerereSQL);
     while($rand = mysql_fetch_array($rezultat)) {
     echo $rand['nume'];
     }

?>
```

Salvati fisierul si apoi accesati in browser http://localhost/tutorial/extragere_date.php

Observati ca daca ati introdus o singura valoare in campul nume din tabela formular, aceasta se va afisa pe pagina, daca sunt mai multe intrari ele se vor afisa una in continuarea celeilalte.
Pentru a le afisa una sub alta modificati linia echo $rand['nume']; in echo $rand['nume'].'
';

Si acum sa analizam codul PHP.

Prima linie din script trebuie sa fie **functia require_once**, functie care include, o singura data, in pagina noastra extragere_date.php pagina config.php; adica datele din config.php vor fi transmise in pagina noastra.

$cerereSQL = 'SELECT * FROM `formular`'; - In aceasta linie avem variabila $cerereSQL cu valoarea cererii SQL pentru a extrage datele din tabela formular. Ea se interpreteaza cam asa: SELECTEAZA tot DIN formular.

$rezultat = mysql_query($cerereSQL); - In aceasta linie avem variabila $rezultat cu valoarea functiei **mysql_query**, functie care realizeaza deschiderea conexiunii.

```
while($rand = mysql_fetch_array($rezultat)) {
echo $rand['nume'];
}
```

In aceasta parte de cod observam bucla while, bucla care am invatat cum se foloseste la inceputul tutorialului nostru.

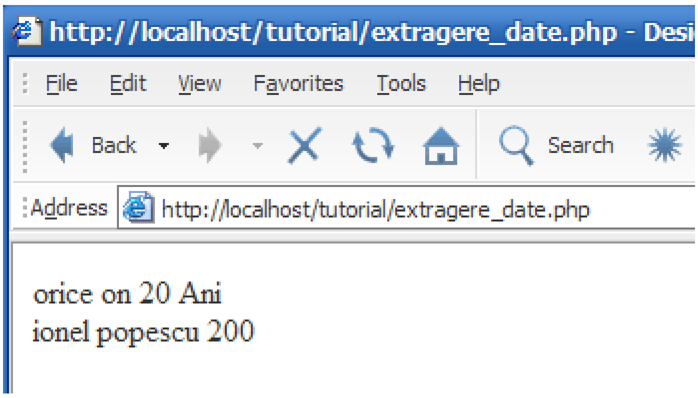
echo $rand['nume']; - In acest echo apar valorile coloanei nume din tabelul formular.

nume
orice
ionel

In interiorul acestei bucle putem afisa si prenumele, si varsta, modificand constructia echo, si anume:

```
echo $rand['nume'].' '.$rand['prenume'].' '.$rand['varsta'].' <br>';
```

Salvati si apoi testati in browser.



In cazul meu, am 2 intrari in tabela formular si anume:

←T→	id	nume	prenume	varsta
	1	orice	on	20 Ani
	2	ionel	popescu	200

In cererea noastra SQL am selectat * adica tot din tabela formular, insa se poate selecta numai un camp sau mai multe.. si anume:

$cerereSQL = 'SELECT `nume,prenume` FROM `formular`';

Sintaxa SELECT este mult mai completa decat am prezentat-o pana acum si anume:

```
SELECT
    [ALL | DISTINCT | DISTINCTROW ]
      [HIGH_PRIORITY]
      [STRAIGHT_JOIN]
      [SQL_SMALL_RESULT] [SQL_BIG_RESULT] [SQL_BUFFER_RESULT]
      [SQL_CACHE | SQL_NO_CACHE] [SQL_CALC_FOUND_ROWS]
    select_expr, ...
    [INTO OUTFILE 'file_name' export_options
      | INTO DUMPFILE 'file_name']
    [FROM table_references
      [WHERE where_definition]
      [GROUP BY {col_name | expr | position}
        [ASC | DESC], ... [WITH ROLLUP]]
      [HAVING where_definition]
      [ORDER BY {col_name | expr | position}
        [ASC | DESC] , ...]
      [LIMIT {[offset,] row_count | row_count OFFSET offset}]
      [PROCEDURE procedure_name(argument_list)]
      [FOR UPDATE | LOCK IN SHARE MODE]]
```

Sa luam urmatorul exemplu, pentru a selecta tot din tabela formular, si sa ii punem o conditie WHERE.

```
<?php
require_once('config.php');

// Selectare dare din baza de date

$cerereSQL = 'SELECT * FROM formular WHERE nume="orice" ';
$rezultat = mysql_query($cerereSQL);
     while($rand = mysql_fetch_array($rezultat)) {
     echo $rand['nume'].' '.$rand['prenume'].' '.$rand['varsta'].' <br>';
     }

?>
```

Salvati si apoi vizualizati in browser.
Observati ca avem conditia WHERE nume="orice", iar rezultatul o sa fie numai prima linie din tabela formular, deoarece primul nume de acolo este egal cu numele dat de noi in conditie.

INSERT

Sintaxa insert se foloseste pentru a adauga date in baza de date.

Sa luam urmatorul exemplu:

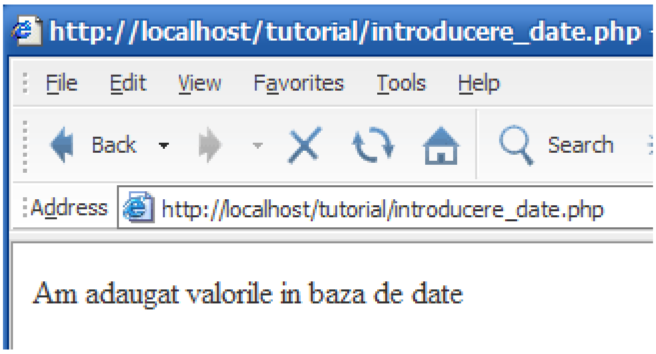
```
<?php
require_once('config.php');

$cerereSQL = "INSERT INTO `formular` (`nume` , `prenume`, `varsta`)
              VALUES ('Ivascu', 'Valentin', '20')";
mysql_query($cerereSQL);

echo 'Am adaugat valorile in baza de date';

?>
```

Creati un fisier nou in folderul tutorial si numiti-l **introducere_date.php**, apoi introduceti codul PHP de mai sus, salvati si vizualizati in browser.



Apoi accesati baza de date prin scriptul phpMyAdmin, intrati la tutorial, dupa care apasati pe tabela formular si sus pe butonul “Navigare”.

Observati ca au fost adaugate datele in tabela formular.

Sintaxa INSERT este mult mai completa decat am prezentat-o pana acum si anume:

```
INSERT [LOW_PRIORITY | DELAYED | HIGH_PRIORITY] [IGNORE]
    [INTO] tbl_name [(col_name,...)]
    VALUES ({expr | DEFAULT},...),(...),...
    [ ON DUPLICATE KEY UPDATE col_name=expr, ... ]
```

sau:

```
INSERT [LOW_PRIORITY | DELAYED | HIGH_PRIORITY] [IGNORE]
    [INTO] tbl_name
    SET col_name={expr | DEFAULT}, ...
    [ ON DUPLICATE KEY UPDATE col_name=expr, ... ]
```

sau:

```
INSERT [LOW_PRIORITY | HIGH_PRIORITY] [IGNORE]
    [INTO] tbl_name [(col_name,...)]
    SELECT ...
    [ ON DUPLICATE KEY UPDATE col_name=expr, ... ]
```

UPDATE

Sintaxa update se foloseste pentru a modifica datele existente din baza de date.

Sa luam urmatorul exemplu:

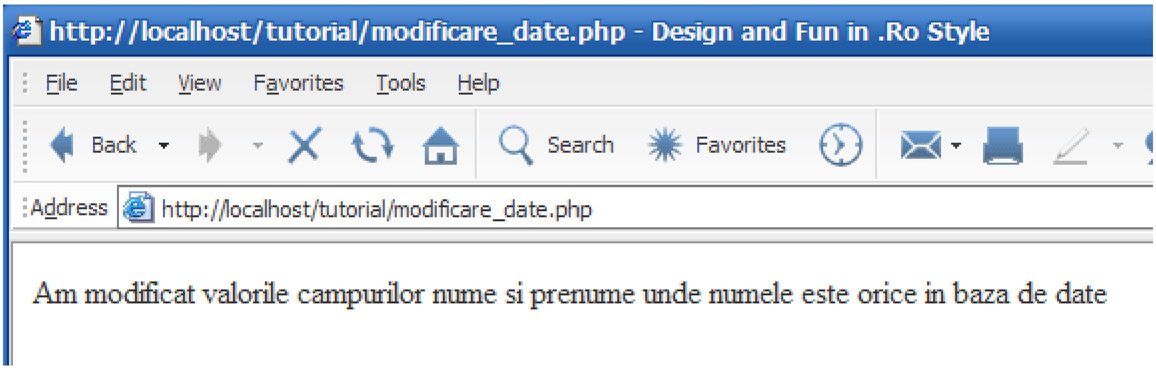
```php
<?php
require_once('config.php');

$cerereSQL = "UPDATE `formular` SET nume='nume', prenume='prenume' WHERE nume='orice' ";
mysql_query($cerereSQL);

echo 'Am modificat valorile campurilor nume si prenume unde numele este orice in baza de date';

?>
```

Creati un fisier nou in folderul tutorial, numiti-l **modificare_date.php** si introduceti codul PHP de mai sus, dupa care salvati si vizualizati in browser.



Accesati baza de date prin phpMyAdmin si observati ca prima coloana a fost modificata.

Sintaxa UPDATE este mult mai completa decat am prezentat-o pana acum si anume:

Sintaxa simpla

```
UPDATE [LOW_PRIORITY] [IGNORE] tbl_name
    SET col_name1=expr1 [, col_name2=expr2 ...]
    [WHERE where_definition]
    [ORDER BY ...]
    [LIMIT row_count]
```

Sintaxa multipla

```
UPDATE [LOW_PRIORITY] [IGNORE] table_references
    SET col_name1=expr1 [, col_name2=expr2 ...]
    [WHERE where_definition]
```

DELETE

Sintaxa delete se foloseste pentru a sterge datele existente din baza de date.

Sa luam urmatorul exemplu:

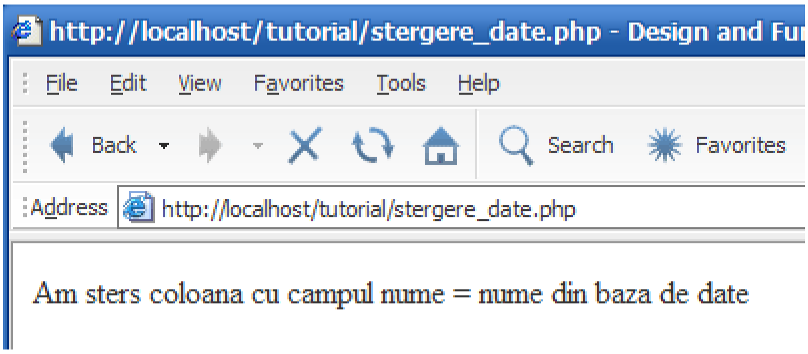
```
<?php
require_once('config.php');

$cerereSQL = "DELETE FROM `formular` WHERE nume='nume'";
mysql_query($cerereSQL);

echo 'Am sters coloana cu campul nume = nume din baza de date';

?>
```

Creati un fisier nou in folderul tutorial si numiti-l **stergere_date.php,** traduceti codul PHP de mai sus, salvati si vizualizati in browser.



Accesati baza de date prin phpMyAdmin si observati ca prima coloana a fost stearsa.

Sintaxa DELETE este mult mai completa decat am prezentat-o pana acum si anume:

Sintaxa simpla

```
DELETE [LOW_PRIORITY] [QUICK] [IGNORE] FROM tbl_name
       [WHERE where_definition]
       [ORDER BY ...]
       [LIMIT row_count]
```

Sintaxa multipla

```
DELETE [LOW_PRIORITY] [QUICK] [IGNORE]
       tbl_name[.*] [, tbl_name[.*] ...]
       FROM table_references
       [WHERE where_definition]
```

sau:

```
DELETE [LOW_PRIORITY] [QUICK] [IGNORE]
       FROM tbl_name[.*] [, tbl_name[.*] ...]
       USING table_references
       [WHERE where_definition]
```

Securitatea scripturilor

Regula numarul unu a securitatii online este: nu va incredeti niciodata in utilizator.
Intotdeauna verificati datele trimise catre server si "curatati-le" inainte de a le utiliza. Pentru aceasta trebuie sa luati in considerare cateva posibile "gauri":

1. Variabilele globale

Avand variabilele globale ON, puteti accesa datele trimise prin formulare mai simplu: $variabila in loc de $_GET['variabila'], la fel si pentru POST sau FILES. Daca scriptul nu este foarte bine gandit, variabilele globale pot prezenta un risc major de securitate. Din acest motiv, **php.ini** este distribuit cu **globals=Off** in ultimele versiuni.

Ca sa fiti siguri ca variabilele globale sunt OFF, intrati in C:\Program Files\EasyPHP-1.8\apache (daca aveti instalat pachetul EasyPHP, daca nu, intrati in C:\WINDOWS\php.ini) apoi deschideti fisierul de configurare **php.ini**, cautati linia register_globals si in cazul in care este setata On, setati-o **register_globals = Off**

Salvati si reporniti serverul apache.

Atentie! Ca sa fiti siguri ca editati fisierul de configurare php care trebuie, realizati o pagina cu numele **php.php** in care introduceti codul de mai jos, apoi testati in browser si observati linia:

Configuration File (php.ini) Path	C:\Program Files\EasyPHP-1.8\apache\php.ini

```
<?php

phpinfo();

?>
```

2. Ghilimelele magice

La inceputul tutorialului am discutat despre ghilimelele magice simple **'** cat si despre cele duble **"**. Acum este momentul sa ne aducem aminte de ele si sa aruncam o privire asupra problemelor ce pot aparea.

O prima problema ar fi aceea de a nu mai face functionala comanda INSERT; sa luam ca exemplu urmatorul cod si sa il analizam.

Realizati un folder cu numele **securitate** si in el creati 2 fisiere - **config.php** cu conectarea la baza de date si **ghilimele_magice.php** si introduceti urmatorul cod:

```
<?php
require_once('config.php');

if(!isset($_GET['pagina'])) $_GET['pagina'] = '';
switch($_GET['pagina']) {

case '':
echo '<form name="securitate" action="ghilimele_magice.php?pagina=insert" method="post">
       Nume: <input type="text" name="nume" value=""> <input type="submit" name="Trimite"
       value="Trimite"></form>';
break;

case 'insert':
if($_POST['nume'] == '') {
echo 'Introdu un nume';
```

```
} else {

$cerereSQL = "INSERT INTO `intrari` (`nume`) VALUES ('".$_POST['nume']."')";
mysql_query($cerereSQL);

echo 'Am introdus numele '.$_POST['nume'].' ';
}

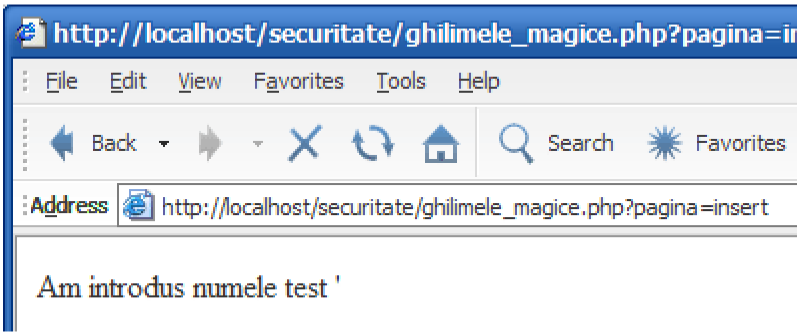
break;

}

?>
```

In acest exemplu, m-am folosit de o baza de date pe care am creat-o pentru exercitiul 1 din acest tutorial. Puteti modifica accest script si sa inlocuiti “intrari” cu un nume al unei tabele din baza dumneavoastra de date, la fel si in fisierul config.php pe care l-ati creat, trebuie sa efectuati modificarile cu numele bazei dumneavoastre de date.

Accesati pagina si introduceti in formular textul test ‘



Observati ca scriptul nostru ne printeaza textul din conditia else.
Accesati baza de date si verificati intrarile; veti observa defapt ca textul test ‘ introdus de dumneavoastra in formular, nu a fost introdus in baza de date.

Acum va veti intreba, de ce?

Uitandu-ne peste codul nostru din pagina ghilimele_magice.php, in case-ul insert observam comanda SQL:

```
$cerereSQL = "INSERT INTO `intrari` (`nume`) VALUES ('".$_POST['nume']."')";
```

Aruncand o privire atenta asupra comentzii SQL, asupra alcatuirii variabilei, constatam faptul ca sunt scrise corect:

1. Variabila este deschisa si inchisa de ghilimelele magice duble $cerereSQL = "";
2. Comanda SQL de insert, este scrisa corect: insereaza in tabela intrari, coloana nume, valoarea postata prin formular INSERT INTO `intrari` (`nume`) VALUES ('".$_POST['nume']."')

Daca este scrisa corect, serverul mysql merge, atunci de ce nu merge?

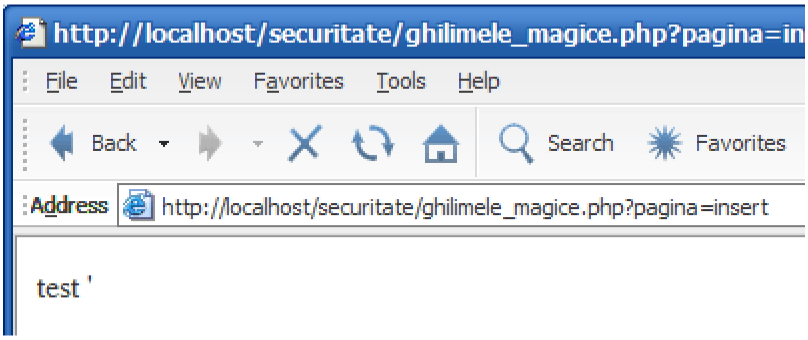
Ca o mica pauza, haideti sa vedem daca datele sunt trimise de catre formular (daca nu erau trimise, se insera valoare goala in baza de date, adica se insera un rand nou gol, insa pentru a fi siguri ca totul merge ok, este bine sa verificam si daca formularul functioneaza):

Comentati variabila $cerereSQL adaugand // in fata ei, mysql_query si echo, apoi cu o linie mai sus scrieti echo $_POST[‘nume’];

```
echo $_POST['nume'];
//$cerereSQL = "INSERT INTO `intrari` (`nume`) VALUES ('".$_POST['nume']."')";
//mysql_query($cerereSQL);

//echo 'Am introdus numele '.$_POST['nume'].' ';
```

Testati si vedeti daca valoarea este aratata in pagina.

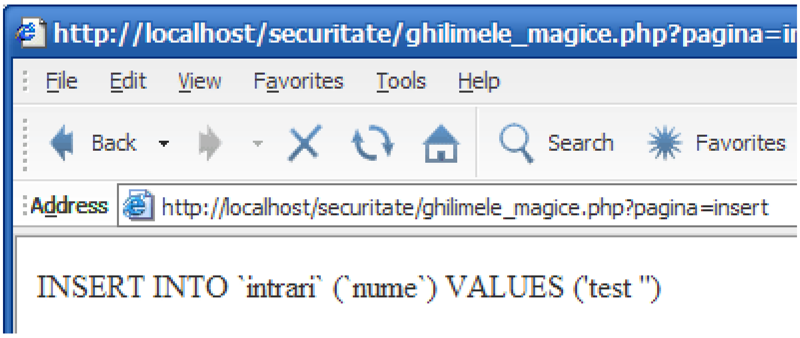


Acum ca am observat valoarea introdusa in formular, stergeti linia `echo $_POST['nume'];` si comentariile adaugate urmatoarelor 2 linii, apoi scrieti un echo inaintea variabilei `$cerereSQL` pentru a vizualiza in pagina comanda care se trimite catre baza noastra de date:

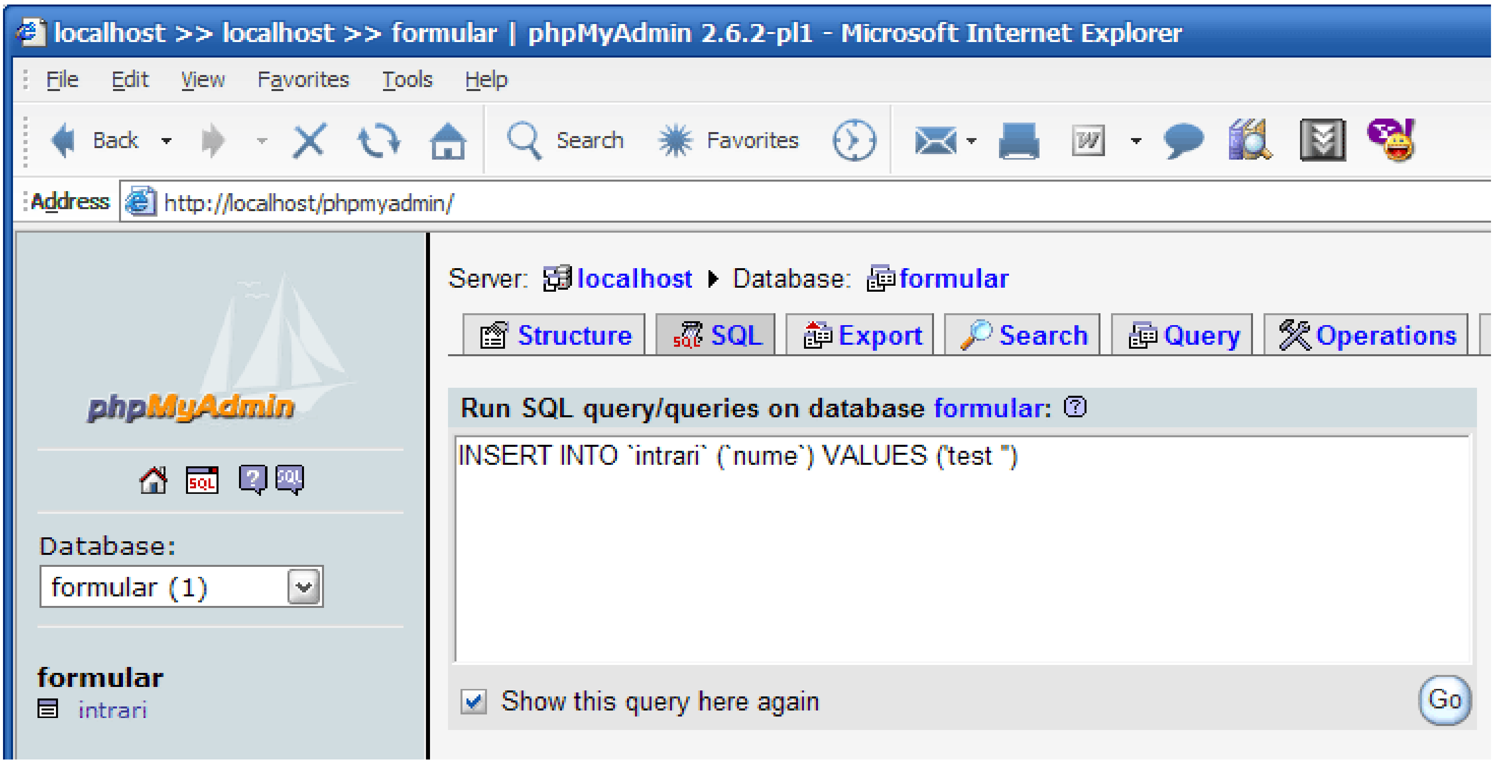
```
echo $cerereSQL = "INSERT INTO `intrari` (`nume`) VALUES ('".$_POST['nume']."')";
mysql_query($cerereSQL);

//echo 'Am introdus numele '.$_POST['nume'].' ';
```

Salvati si apoi accesati in browser



Copiati acest rezultat, apoi accesati scriptul phpMyAdmin http://localhost/phpmyadmin , navigati pana la baza de date cu care lucrati, dati click pe ea si sus de tot observati meniul. Apasati pe SQL, apoi in pagina rezultata dati paste la linia pe care ati copiat-o din pagina noastra.



Apasati pe butonul Go, apoi sa vedeti daca comanda se executa cum trebuie, direct din scriptul phpMyAdmin, si se insereaza acea valoare in baza de date:

Error

There seems to be an error in your SQL query. The MySQL server error output below, if there is any, may also help you in diagnosing the problem

```
ERROR: Unclosed quote @ 39
STR: '
SQL: INSERT INTO `intrari` (`nume`) VALUES ('test '')
```

SQL query:

INSERT INTO `intrari` (`nume`) VALUES ('test '')

MySQL said:

```
#1064 - You have an error in your SQL syntax; check the manual that corresponds to your
MySQL server version for the right syntax to use near ''test '')' at line 1
```

[Back]

UPS! Se pare ca am primit o eroare SQL.

Citind si traducand eroarea, ne spune cam asa:

```
ERROR: Ghilimea magica neinchisa @ 39
STR: '
SQL: INSERT INTO `intrari` (`nume`) VALUES ('test '')
```

Dupa ce am luat o pauza cu depanarea si depistarea problemei din scriptul nostru, ne intoarcem la problema de unde am plecat.

Se pare ca a inceput sa se faca lumina in cazul nostru, si anume.. observam faptul ca a intervenit o eroare in comanda noastra SQL din cauza valorii trimise prin post.

Valoarea trimisa prin post continea o ghilimea magica simpla, iar comanda SQL la fel:

```
VALUES ('".$_POST['nume']."')";
```

Adaugand valoarea din post, comanda SQL devenea: `VALUES ('test '')";` adica, o ghilimea simpla neinchisa, o ghilimea simpla in plus (aducandu-ne aminte de primele lectii de la inceputul tutorialului, am discutat ca in scripturile PHP, o ghilimea magica deschisa.. trebuie sa fie si inchisa).

In concluzie, scriptul nu a mai mers din cauza conflictelor aparute din cauza unei ghilimele simple.

Putin mai incolo va voi explica si cum puteti sa evitati aceasta problema, insa, acum, haideti sa luam acelasi exemplu, doar ca vom modifica putin scriptul, si anume linia cu cererea SQL:

Inlocuiti linia `$cerereSQL = "INSERT INTO `intrari` (`nume`) VALUES ('".$_POST['nume']."')";`
cu linia `$cerereSQL = 'INSERT INTO `intrari` (`nume`) VALUES ("'.$_POST['nume'].'")';`

Dupa cum vedeti, am schimbat variabila `$cerereSQL`, inlocuind ghilimelele magice, astfel incat, in paranteza de la `VALUES`, vom avea ghilimelele magice duble.

Testati scriptul din nou introducand aceeasi valoare, si anume test ‘, dupa care verificati daca a fost adaugata in baza de date



Dupa cum vedeti, comanda SQL a fost executata, si valoarea a fost introdusa in baza de date.

Uitandu-ne la VALUES din comanda SQL, vedem ca ghilimeaua simpla ‘ nu mai face conflict cu cele duble “

```
VALUES ("test '")
```

Daca in formular introducem valoarea test “ , ne vom lovi de aceeasi problema ca cea anterioara, adica nu se va executa comanda SQL:

```
VALUES ("test "")
```

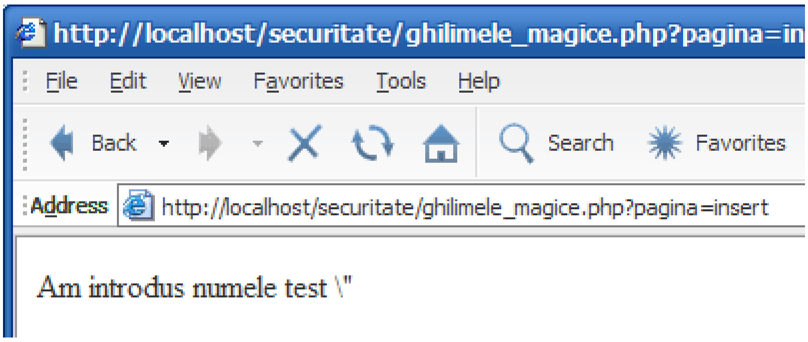
Observam conflictul intre ghilimeaua magica dubla si cele de delimitare continut din interiorul parantezei.

Pentru a scapa de aceasta problema neplacuta, avem mai multe posibilitati:

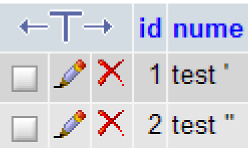
1) magic_quotes_gpc – Intrati in **php.ini** si verificati daca magic_quotes_gpc este On, daca nu, setati-l dumneavoastra On. (magic_quotes_gpc = On)

Salvati, opriti si reporniti serverul apache.
Aceasta optiune adauga automat **** atunci cand detecteaza o ghilimea magica simpla sau dubla, devenind astfel din **‘** in **\’** respectiv din **“** in **\”** (sa ne amintim faptul ca am invatat la inceputul tutorialului despre anularea unei ghilimele folosind linia inversa \ ).

Testati scriptul din nou adaugand pe rand atata valoarea test ‘, cat si valoarea test “, apoi verificati baza de date



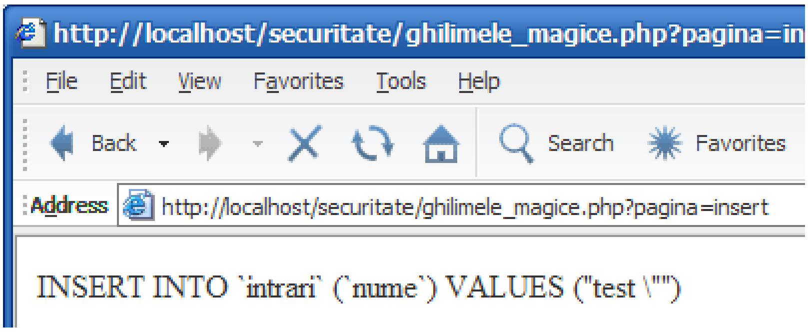
Observam ca atat test ‘ cat si test “ au fost adaugate in baza de date



Ca sa intelegem si mai bine, comentati linia cu echo din script si puneti un echo in fata la variabila `$cerereSQL`

```
echo $cerereSQL = 'INSERT INTO `intrari` (`nume`) VALUES ("'.$_POST['nume'].'")';
mysql_query($cerereSQL);

//echo 'Am introdus numele '.$_POST['nume'].' ';
```



Observati cum ghilimeaua magica dubla a fost anulata (pentru scriptul PHP) si cum scriptul devine astfel corect.

Inainte de a trece la punctul 2 din lista posibilitatilor de protectie asupra ghilimelelor magice, intrati in php.ini si setati **magic_quotes_gpc = Off**

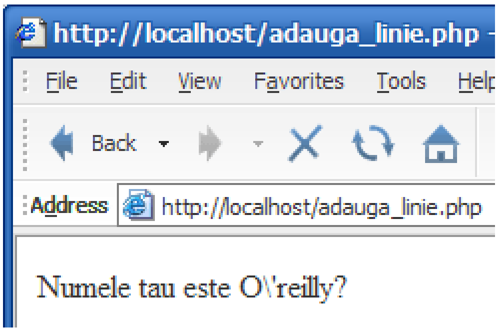
2) addslashes(); - Aceasta este o functie PHP ce actioneaza la fel ca setarea php.ini explicata mai sus, adaugand **** in fata ghilimelelor magice simple sau duble.

Realizati o pagina cu numele **adauga_linie.php**, introduceti codul de mai jos, apoi testati in browser.

```
<?php

$text = "Numele tau este O'reilly?";
echo addslashes($text);

?>
```



Observati ca actioneaza la fel ca magic_quotes_gpc

3) mysql_real_escape_string(); - Este o functie MySQL care, la fel ca si celelalte posibilitati de protectie descrise mai sus, adauga o linie inversa **** in fata caracterelor speciale: NULL (ASCII 0), \n, \r, \, ', ", \x1a

Utilizand acelasi exemplu de script ca si cel de la numarul 1, modificati variabila `$cerereSQL` in:

```
$cerereSQL = 'INSERT INTO `intrari` (`nume`)
              VALUES ("'.mysql_real_escape_string($_POST['nume']).'")';
```

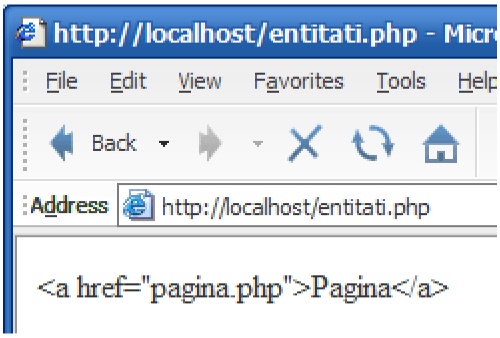
Salvati, apoi completati pe rand formularul introducand valorile test ‘, respectiv test “

Testati in browser si verificati in baza de date daca au fost introduse.
Observati ca au fost introduse, ca si mai sus, fara nici o problema.

4) htmlentities(); - Este o functie PHP care transforma caracterele speciale HTML in entitati ale acestora.

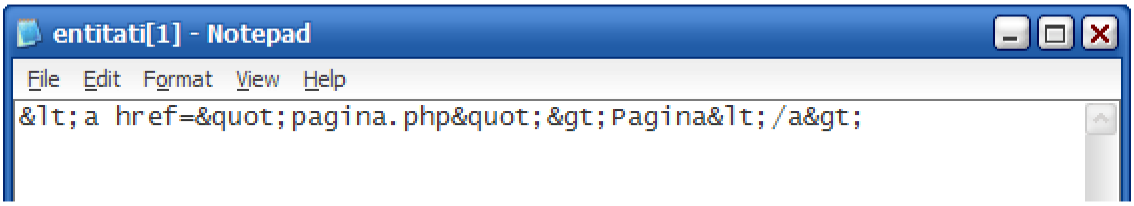
Realizati o pagina cu numele **entitati.php**, introduceti codul de mai jos si testati in browser:

```
<?php
$cod_html = '<a href="pagina.php">Pagina</a>';
echo htmlentities($cod_html);
?>
```



Observati faptul ca browserul nu mai interpreteaza codul HTML ci il afiseaza in pagina.

Vizualizati sursa apasand View -> Source:



Dupa cum vedeti, functia noastra htmlentities(); ne-a convertit caracterele speciale HTML si ghilimelele in coduri ale acestora:

< a devenit &alt;
" a devenit "
> a devenit >

Intorcandu-ne la scriptul folosit la punctul 1, modificati variabila $cerereSQL in:

```
$cerereSQL = 'INSERT INTO `intrari` (`nume`)
              VALUES ("'.htmlentities($_POST['nume'], ENT_QUOTES).'")';
```

Salvati si completati pe rand formularul, introducand valorile test ', respectiv test "

Testati in browser apoi verificati si in baza de date daca au fost introduse.

←T→	id	nume
	1	test "
	2	test '

Observati faptul ca atat ghilimeaua magica simpla, cat si cea dubla a fost codata si introdusa in baza de date fara nici o problema.

Uitandu-ne in interiorul cererii noastre SQL, observam faptul ca functia htmlentities este urmata de un parametru.

Aceasta functie are 3 parametrii pentru a coda ghilimelele magice:

ENT_COMPAT - Converteste numai ghilimelele magice duble.
ENT_QUOTES - Converteste numai ghilimelele magice simple.
ENT_NOQUOTES - Nu converteste nici o ghilimea.

Acum, pe acelasi exemplu, introduceti in formular valoarea test \, apoi verificati daca a fost adaugata in baza de date.

Observati ca nu!

De ce?

Aceasta functie nu converteste si linia inversa ****, fapt ce va duce la anularea ghilimelei din interiorul cererii SQL, facand cererea SQL gresita.

Puneti un echo in fata variabilei `$cerereSQL` si testati in browser, introducand aceeasi valoare in formular.



Pentru a rezolva aceasta problema, am realizat o noua functie PHP in care am codat si \

```
<?php

function addentities($data){
   if(trim($data) != ''){
   $data = htmlentities($data, ENT_QUOTES);
   return str_replace('\\', '&#92;'), $data);
   } else return $data;
} // End addentities() --------------

?>
```

Adaugati aceasta functie in pagina **ghilimele_magice.php** dupa linia `require_once('config.php');`
si modificati variabila `$cerereSQL` in:

```
$cerereSQL = 'INSERT INTO `intrari` (`nume`) VALUES ("'.addentities($_POST['nume']).'")';
```

Salvati si testati din nou in browser, introducand in formular valoarea test \ , apoi verificati daca a fost introdusa in baza de date.

←T→	id	nume
	1	test \

Observati ca, folosind **NOUA** functie, de acum si linia inversa va fi codata si vom scapa de probleme.

Pana acum am vorbit si am dat exemple de protectie pentru inserarea in baza de date, insa sa stiti ca aceleasi exemple se aplica si atunci cand realizati o selectare din baza de date, cat si in momentul in care stergeti date sau orice alta operatiune cu baza de date.

Ca exemplu, sa presupunem ca interogarea SQL de verificare a numelui si a parolei pentru inregistrarea pe site este SELECT * FROM users WHERE nume='$nume' AND parola='$parola' si daca interogarea este executata cu succes, utilizatorul este logat. In acest caz, folosind parola 'OR' 1=1, oricine poate avea acces pe site, deoarece interogarea SELECT * FROM user WHERE nume='un nume oarecare' AND parola='' OR '1=1' este executata cu succes si returneaza un rezultat (toate inregistrarile din baza de date, de fapt).

Aceste operatiuni sunt intalnite sub numele de **SQL Injection**

Sa luam urmatorul exemplu, in care dumneavoastra aveti 2 pagini PHP: in prima aveti o un link prin care transferati cu $_GET id-ul unui utilizator, iar in pagina a-II-a aveti o selectare dupa ID in functie de valoarea $_GET trimisa din prima pagina:

Pagina 1 o sa arate cam asa:

```
cerere sql
while() {
echo '<a href="pagina2.php?id='.$rand['id'].'">'.$rand['nume'].'</a><br>';
}
```

Va sunt listate numele din baza de date unul sub altul, iar ca link veti avea calea catre pagina 2 si $_GET['id'] cu valoarea fiecarui id atribuit fiecarei inregistrari (fiecarui user) din tabela data.

In momentul in care dati click pe un nume, veti fi transferat in pagina2, unde veti avea un nou select insa de data asta in functie de ID-ul pasat din prima pagina.

```
$cerereSQL = 'SELECT * FROM `intrari` WHERE id="'.$_GET['id'].'"';
```

Revenind la problema de unde am plecat, securitatea, va rog sa va uitati asupra acestei parti si sa meditati putin.

Valoarea pasata de $_GET['id'] nu poate sa fie decat numerica, deoarece ID-ul este NUMAI numeric.
In acest caz, la inceputul paginii 2.php veti seta urmatorul cod:

```
if(!is_numeric($_GET['id'])) {
echo 'ID nu este numeric, ce incerci sa faci ?';
} else {
$cerereSQL = 'SELECT * FROM `intrari` WHERE id="'.$_GET['id'].'"';
..
..
}
```

Daca valoarea trimisa prin $_GET['id'] nu este numerica, punem in script un text care sa-l afiseze in pagina, daca este numerica, atunci scriptul executa comanda SQL.

Daca si dumneavoastra ati gandit asa, inseamna ca sunteti pe drumul cel bun in ceea ce priveste securitatea.

Nota: Primele 3 modalitati de protectie se folosesc atunci cand doriti sa reutilizati in pagina, codul introdus in baza de date, iar modalitatea 4 cu functia addentities(); o puteti folosi atunci cand doriti sa protejati pagina atat de codurile HTML introduse in formular, cat si de caracterele speciale PHP.
Eu unul va recomand aceasta optiune, insa este la alegerea dumneavoastra care din ele le veti folosi (dupa cum am spus.. in functie si de nevoile de folosire).

3. Includere

Incercati sa evitati includerea "vizibila" a fisierelor in forma http://localhost/pagina.php?fisier=cauta.html pentru a include fisiere in cadrul unei pagini. Cu putina neatentie din partea voastra, atacatorul ar putea accesa astfel de informatii sensibile din cadrul sistemului. Nu includeti fisiere straine. PHP poate "include" fisiere aflate pe alte servere decat cel care ruleaza daca setarea URL fopen wrappers este activata in php.ini.

In exemplul de mai sus, un atacator ar fi putut accesa adresa http://localhost/pagina.php?fisier=http://www.rau.ro/scriptultau.php pentru a include in fisier un script localizat pe alt server si astfel obtine acces catre toate resursele sistemului la care are acces PHP, putand rula comenzi de sistem, afisa informatii confidentiale sau sterge baza de date. Nu permiteti includerea fisierelor din alta parte decat de pe serverul vostru.

Setati **allow_url_fopen = Off** in **php.ini**.

4. Formulare

Folositi metoda **POST** in formulare atunci cand informatia din acestea urmeaza sa fie introdusa in baza de date. Daca variabilele globale sunt OFF in php.ini sau metoda de transmitere a formularului este GET, un utilizator rau intentionat ar putea accesa adresa dvs. si ar introduce comentariul lui (exemplu "blabla") in baza voastra de date, fara sa treaca propriuzis prin site. Asa va puteti trezi ca nu mai aveti spatiu pe server, iar in baza de date sunt cateva milioane de comentarii care spun acelasi lucru "blabla".

Verificarea provenientei cererilor catre server este foarte importanta si in alt caz: formularele de loghin. Cineva care stie numele de utilizator ar putea incerca sa va gaseasca parola foarte usor. In acest caz, va trebui sa puneti o protectie suplimentara care sa nu permita mai mult de trei incercari consecutive esuate de logare pentru un nume de utilizator.

Aceasta problema se rezolva usor folosind sesiunile. In momentul cand cineva incearca sa trimita numele si parola din formular, putem seta o variabila de sesiune $_SESSION['login_count'] care sa tina minte numarul de incercari.

Cand valoarea acesteia trece de 3 (incercari nereusite) nici macar nu mai interogati baza de date pentru a verifica a patra incercare. Variabila de sesiune va ramane in memorie cat timp browserul este deschis si sesiunea activa (optiunea implicita a PHP de mentinere a sesiunilor active este de o ora). Daca atacatorul asteapta o ora sau isi inchide browserul, sesiunea va fi inchisa si va putea de alte trei ori sa se logheze. De cele mai multe ori acesta masura de siguranta este suficienta pentru a preveni incercarile de aflare a parolelor.

5. Extensii

O practica obisnuita este de a acorda extensia **.inc** fisierelor care contin biblioteci de functii ce urmeaza a fi incluse si folosite in cod. PHP nu parseaza fisierele cu extensia .inc si daca acestea sunt apelate direct ele sunt trimise plain text catre browser. Nu puneti informatii sensibile (precum nume si parola) in fisiere cu extensia .inc, .txt, sau .html care pot fi accesate si vazute. Folositi pentru aceste fisiere extensia .php care, daca sunt accesate direct, vor fi rulate fara sa afiseze informatiile continute in ele.

6. Comanda CHMOD – setarea atributelor

Comanda CHMOD (change access permissions of a file) este folosita pentru a schimba modul de acces (de permitere) a fisierelor si directoarelor de pe server.

Setarea poate afecta modul cum poate fi citit, sau cum se poate executa un fisier pe server.
De exemplu, daca aveti un fisier .php care trebuie sa execute o comanda de scriere pe server intr-un fisier .txt trebuie sa ii dati dreptul sa poata fi executat iar la fisierul .txt trebuie sa ii dati dreptul de a se putea scrie in el.

Aveti 3 tipuri de acces - OWNER - GROUP - WORLD - fiecare cu cate 3 setari - READ - WRITE - EXECUTE –

Este bine ca la WORLD sa nu dati decat acces la citire (pentru a putea accesa paginile de pe site) dar nu si drepturi de scriere sau executie (pentru ca nu doriti ca oricine sa va scrie ce vrea in respectivul firier .txt)
In programul WINDOWS COMMANDER (pe care eu il foosesc si ca FTP) setarea atributelor pe server se face din meniul FILES - CHANGES ATTRIBUTES. Fiecare program de FTP are in meniul lui setare pentru atributele fisierelor de pe server.

7. SESIUNI

Daca aveti o sectiune de administrare pe site, unde accesul este restrictionat doar la membrii de exemplu, este bine ca in toate paginile din aceasta sectiune sa includa o pagina de verificare a accesului.

Aceasta pagina va verifica la fiecare accesare daca utilizatorul este inregistrat si are acces pe pagina respectiva si permite rularea paginii, doar daca utilizatorul este inregistrat. Fara aceasta verificare, un utilizator ar putea accesa paginile din sectiunea de administrare fara sa treaca prin formularul de inregistrare.

Dupa autentificarea propriu-zisa, vom folosi variabile de sesiune pentru a pastra in memorie cateva informatii despre autentificare, pentru a le verifica mai tarziu, atunci cand accesam alte pagini din cadrul sectiunii de administrare. Pornim intai sesiunea dupa care trecem la salvarea informatiilor in ea ca in exemplul de mai jos:

```
session_start();
$_SESSION['nume_admin'] = $_POST['nume'];
$_SESSION['parola_encriptata'] = $parolaEncriptata;
```

Pe langa acestea, pentru o si mai mare siguranta, vom salva id-ul sesiunii in alta variabila. Toate sesiunile au un id unic, un string care seamana cu rezultatul unei criptari MD5:

```
$_SESSION ['key_admin'] = session_id();
```

Cu autentificare facuta, spunem scriptului sa incarce prima pagina din sectiunea de administrare:

```
header("location: admin.php");
```

In continuare, pentru a impiedica accesul neautorizat la paginile din aceasta sectiune, scriem un mic script de verificare a datelor sesiunii inainte de a incarca orice pagina din sectiunea de administrare:

```
session_start();
if($_SESSION['key_admin'] != session_id()) {o
echo 'Acces neautorizat!';
exit;
}
```

8. Loguri de acces

Pentru o verificare ulterioara a persoanelor care incearca sa intre in paginile restrictionate de pe site, este bine sa salvam in baza de date (sau intr-un fisier text) numele, parola criptata, ora, data si ip-ul. Asa veti stii cine a incercat sa va sparga site-ul si puteti crea un script care sa verifice de exemplu ip-ul si daca corespunde cu unul din cele care sunt blocate sa nu aiba acces la nici una din pagini (chiar daca gaseste userul si parola corecte).

9. Criptarea parolelor cu ajutorul functiei MD5();

Parolele care sunt salvate in baza de date este recomandat sa le tineti criptate, astfel daca cineva incearca sa citeasca parolele le va vedea criptate.

```
$cerereSQL = "INSERT INTO `admin` (`user` , `parola`)
              VALUES ('oriceon', '".md5($_POST['parola'])."')";
```

Criptarea folosind **md5**, teoretic, nu este reversibila (si astfel nici dvs., nici altcineva nu o va putea afla chiar daca are acces la baza de date).

10. Pagina index.html in subdirectoare

Daca aveti mai multe directoare si subdirectoare pe server, este recomandat, pentru a preveni accesul la datele aflate in aceste directoare, sa introduceti in fiecare o pagina numita **index.html** sau **index.php** care sa faca redirectarea automat catre pagina principala a site-ului sau pur si simplu sa nu aibe continut.

Astfel, minimizati riscul sa intre cineva si sa vada tot ce este in aceste subdirectoare.

11. Fisierul .htaccess

Pentru a bloca accesul la un anumit director (sau chiar la tot site-ul), puteti crea un fisier numit .htaccess (cu punct inainte), in care sa introduceti ip-ul care doriti sa il blocati (sau care sa aiba acces)

```
Order Deny,Allow
# Deny from all
Allow from all
```

In exemplul de mai sus, toata lumea are acces. Daca de exemplu ip-ul 192.168.1.1 nu doriti sa aiba acces la acel director, scrieti:

```
Deny from 192.168.1.1
```

Atentie, sa nu va blocati singuri ip-ul la site, ca altfel trebuie sa luati legatura cu administratorul serverului sa stearga fisierul ca sa puteti intra din nou pe site. (la subdirectoare stergeti directorul sau salvati un alt fisier .htaccess) In cazul in care dupa de ati pus fisierul pe server nu il vedeti, el este acolo dar serverul seteaza acest nume de fisier ca hiden.

12. Robotii de cautare (Google, Yahoo . .)

Motoarele de cautare pot indexa tot ce se afla la voi pe site.

Daca doriti ca un fisier sau director sa nu fie indexat de catre motoarele de cautare, trebuie sa creati un fisier numit **robots.txt** care sa se afle in directorul principal de pe site (nu in subdirectoare). In el scrieti urmatoarele comenzi:

\# **robots.txt for http://www.oriceon.com**
User-agent: *
Disallow: /tutorial/
Disallow: /admin
Disallow: /search

Infrumusetarea codului PHP

Este bine ca atunci cand lucrati, sa va creati un stil cat mai frumos, sa nu lucrati cu multe spatii (in toate exemplele am lasat spatiile intentionat pentru a fi mai usor vizibil corul incepatorilor).

Sa luam un exemplu de functie, apoi sa o infrumusetam si sa observam rezultatele vizuale:

```php
<?php
function addentities($data){
if(trim($data) != ''){
$data = htmlentities($data, ENT_QUOTES);
return str_replace('\\', '&#92;'), $data);
} else return $data;
} // End addentities() --------------
?>
```

Si acum sa o infrumusetam folosind spatii (in editorul dvs, folositi TAB-ul):

```php
<?php

function addentities($data){
   if(trim($data) != ''){
   $data = htmlentities($data, ENT_QUOTES);
   return str_replace('\\', '&#92;', $data);
   } else return $data;
} // End addentities() --------------

?>
```

Exercitiul 1

Realizarea unui formular de prelucrare a datelor, introducerea informatiilor in baza de date si vizualizarea acestora

Problema:

Se cer urmatoarele campuri: nume (tipul text), prenume (tipul text), varsta (tipul text), email (tipul text) si comentariu (textarea).

Cerinta:

Toate campurile sa fie obligatorii, campurile nume si prenume trebuie sa contina numai litere, campul varsta sa contina numai cifre, iar in campul comentariu sa nu poata fie introduse mai mult de 255 caractere.

Rezolvare:

Prima si prima oara trebuie sa realizam baza de date.
Intram in phpMyAdmin si cream o baza de date cu numele formular apoi tabela intrari cu 6 coloane si anume:

id | nume | prenume | varsta | email | comentariu

(Dupa cum am invatat mai sus in capitolul MySQL, campul ID trebuie sa fie de tipul INT, auto_increment si primary, apoi campurile: nume CHAR(60), prenume CHAR(60), varsta CHAR(10), email CHAR(100), comentariu CHAR(255)).

Table intrari has been created.

SQL query:

```
CREATE TABLE `intrari` (
  `id` INT NOT NULL AUTO_INCREMENT ,
  `nume` CHAR( 60 ) NOT NULL ,
  `prenume` CHAR( 60 ) NOT NULL ,
  `varsta` CHAR( 10 ) NOT NULL ,
  `email` CHAR( 100 ) NOT NULL ,
  `comentariu` CHAR( 255 ) NOT NULL ,
  PRIMARY KEY ( `id` )
) TYPE = MYISAM ;
```

[Edit] [Create PHP Code]

	Field	Type	Collation	Attributes	Null	Default	Extra	Action
	id	int(11)			No		auto_increment	
	nume	char(60)	latin1_swedish_ci		No			
	prenume	char(60)	latin1_swedish_ci		No			
	varsta	char(10)	latin1_swedish_ci		No			
	email	char(100)	latin1_swedish_ci		No			
	comentariu	char(255)	latin1_swedish_ci		No			

Dupa ce ati realizat baza de date, intrati in directorul radacina al serverului dvs (daca folositi EasyPHP este folderul www) si creati un folder cu numele **formular**.

Realizati fisierul cu numele **config.php** unde vom introduce datele de configurare la baza de date.

```
<?php
session_start();
set_time_limit(0);
error_reporting(E_ALL);

// Informatii baza de date

    $AdresaBazaDate = "localhost";
    $UtilizatorBazaDate = "root";
    $ParolaBazaDate = "parola_baza";
    $NumeBazaDate = "formular";

$conexiune = mysql_connect($AdresaBazaDate,$UtilizatorBazaDate,$ParolaBazaDate)
or die("Nu ma pot conecta la MySQL!");
mysql_select_db($NumeBazaDate,$conexiune) or die("Nu gasesc baza de date!");

function addentities($data){
    if(trim($data) != ''){
    $data = htmlentities($data, ENT_QUOTES);
    return str_replace('\\', '&#92;', $data);
    } else return $data;
} // End addentities() --------------

?>
```

Creati un fisier nou in folderul formular si numiti-l **index.php** apoi introduceti codul de mai jos, salvati si vizualizati in browser.

```
<?php
require_once('config.php');

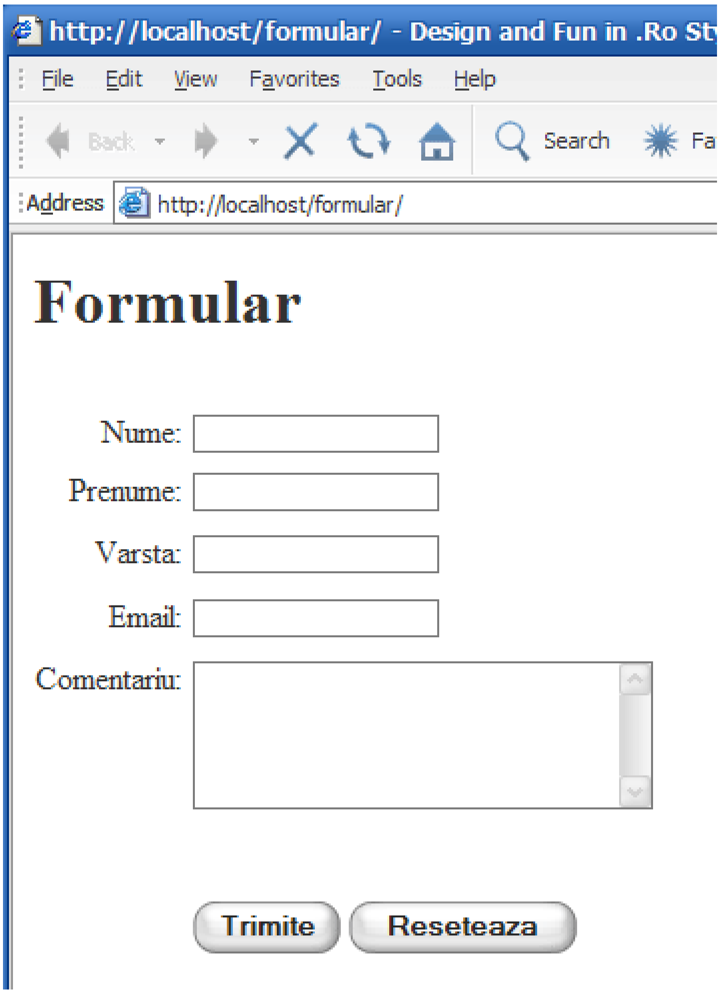
if(!isset($_SESSION['nume'])) $_SESSION['nume'] = '';
if(!isset($_SESSION['prenume'])) $_SESSION['prenume'] = '';
if(!isset($_SESSION['varsta'])) $_SESSION['varsta'] = '';
if(!isset($_SESSION['email'])) $_SESSION['email'] = '';
if(!isset($_SESSION['comentariu'])) $_SESSION['comentariu'] = '';

echo '<table width="310" border="0" cellpadding="0" cellspacing="0">
<form name="formular" action="validare.php" method="post">
  <tr>
    <td height="36" colspan="3" valign="top"><h1>Formular</h1>Comentariul nu trebuie sa
fie mai lung de 255 caractere.</td>
    <td width="1"></td>
   </tr>
   <tr>
     <td width="80" height="19" valign="top"> </td>
     <td width="15" rowspan="10" valign="top"> </td>
     <td width="214" valign="top"> </td>
     <td></td>
   </tr>
   <tr>
     <td height="22" align="right" valign="top">Nume:</td>
     <td valign="top">
       <input type="text" name="nume" value="'.$_SESSION['nume'].'">    </td>
     <td></td>
   </tr>
   <tr>
```

```
      <td height="7"></td>
      <td></td>
      <td></td>
    </tr>
    <tr>
      <td height="22" align="right" valign="top">Prenume:</td>
      <td valign="top"><input type="text" name="prenume"
value="'.$_SESSION['prenume'].'"></td>
      <td></td>
    </tr>
    <tr>
     <td height="9"></td>
      <td></td>
      <td></td>
    </tr>
   <tr>
     <td height="22" align="right" valign="top">Varsta:</td>
      <td valign="top"><input type="text" size="3" maxLength="3" name="varsta"
value="'.$_SESSION['varsta'].'"> ani</td>
      <td></td>
    </tr>
    <tr>
      <td height="10"></td>
      <td></td>
      <td></td>
    </tr>
    <tr>
      <td height="22" align="right" valign="top">Email:</td>
      <td valign="top"><input type="text" name="email"
value="'.$_SESSION['email'].'"></td>
      <td></td>
    </tr>
    <tr>
      <td height="9"></td>
      <td></td>
      <td></td>
    </tr>
    <tr>
      <td height="19" align="right" valign="top">Comentariu:</td>
      <td rowspan="2" valign="top"><textarea name="comentariu" cols="30" rows="5"
value="'.$_SESSION['comentariu'].'">'.$_SESSION['comentariu'].'</textarea></td>
      <td></td>
    </tr>
    <tr>
     <td colspan="2" rowspan="3" valign="top"> </td>
      <td height="83"></td>
    </tr>
    <tr>
      <td height="17" valign="top"> </td>
      <td></td>
    </tr>
    <tr>
      <td height="24" valign="top"><input name="Trimite" type="submit" id="Trimite"
value="Trimite">
        <input name="Reseteaza" type="reset" id="Reseteaza" value="Reseteaza"> </td>
      <td></td>
    </tr>
    </form>
</table>';

?>
```

Testati in browser si observati formularul.



In aceasta pagina avem formularul HTML iar la inceput observati constructia if cu instructiunea isset.

```
if(!isset($_SESSION['nume'])) $_SESSION['nume'] = '';
```

Aceasta linie se interpreteaza asa:

Daca nu este setata sesiunea nume, o setam ca fiind goala

Aceasta operatiune se face pentru a evita afisarea unei erori php ce ne va spune cum ca sesiunea nume nu exista, ceea ce asa este.

Analizand HTML-ul, observam campurile carora le-am atribuit numele corespunzatoare, si anume:

```
...
 <td height="22" align="right" valign="top">Nume:</td>
    <td valign="top">
         <input type="text" name="nume" value="">
    </td>
...
```

Acum ca am realizat formularul, trebuie sa ii prelucram datele prin $_POST si sa verificam continutul acestora, apoi sa il introducem in baza de date.

Realizati un fisier cu numele **validare.php** si introduceti codul de mai jos, apoi testati in browser, introduceti date in formular si apasati butonul “Trimite”.

```php
<?php
require_once('config.php');

$_SESSION['nume'] = $_POST['nume'];
$_SESSION['prenume'] = $_POST['prenume'];
$_SESSION['varsta'] = $_POST['varsta'];
$_SESSION['email'] = $_POST['email'];
$_SESSION['comentariu'] = $_POST['comentariu'];

echo 'Nume: '.$_SESSION['nume'].'<br>
      Prenume: '.$_SESSION['prenume'].'<br>
      Varsta: '.$_SESSION['varsta'].'<br>
      Email: '.$_SESSION['email'].'<br>
      Comentariu: '.$_SESSION['comentariu'].'<br><br>

Daca datele sunt corecte, apasati <a href="prelucrare.php">aici</a> pentru a le valida
<br> si a le introduce in baza de date.';

?>
```

Ca noutate, observati ca am folosit $_SESSION. Pentru mai multe informatii despre sesiuni, si pentru folosirea acestora, va rog sa cititi la pagina 104-105...

Dupa ce am realizat cele doua fisiere, realizati inca unul cu numele **prelucrare.php**, introduceti codul de mai jos si testati din nou in browser.

```php
<?php
require_once('config.php');

if(($_SESSION['nume'] == "") || ($_SESSION['prenume'] == "") || ($_SESSION['varsta'] ==
"") || (!is_numeric($_SESSION['varsta'])) || ($_SESSION['email'] == "") ||
($_SESSION['comentariu'] == "") || (strlen($_SESSION['comentariu']) > 255) )
{
echo 'Nu ai introdus date in formular sau cele introduse nu sunt corecte. <br>
      Apasa <a href="index.php">aici</a> pentru a te intoarce la pagina anterioara.';
} else {
echo 'Va multumim. <br>
      Datele au fost introduse cu succes in baza de date. <br>
      Pentru vizualizare apasati <a href="vizualizare.php">aici</a>.';

$cerereSQL = "INSERT INTO `intrari` (`nume`, `prenume`, `varsta`, `email`,
              `comentariu`)
               VALUES ('".$_SESSION['nume']."', '".$_SESSION['prenume']."',
'".$_SESSION['varsta']."', '".$_SESSION['email']."', '".$_SESSION['comentariu']."');";
mysql_query($cerereSQL);

$_SESSION['nume'] = '';
$_SESSION['prenume'] = '';
$_SESSION['varsta'] = '';
$_SESSION['email'] = '';
$_SESSION['comentariu'] = '';

}

?>
```

Dupa ce ai testat in browser si ati trecut toti pasii din formular, accesati phpMyAdmin si vizualizati datele introduse in baza de date.

←T→	id	nume	prenume	varsta	email	comentariu
	1	orice	on	20	oriceon@yahoo.com	Acesta este un comentariu mai mic de 255 caractere...

Observati ca in scriptul **prelucrare.php** ne-am folosit de cunostintele acumulate in acest tutorial, si anume:

Prima linie, require_once('config.php'); include fisierul config.php in care sunt variabilele cu datele de conectare la baza de date.

In continuare avem conditiile if si else.

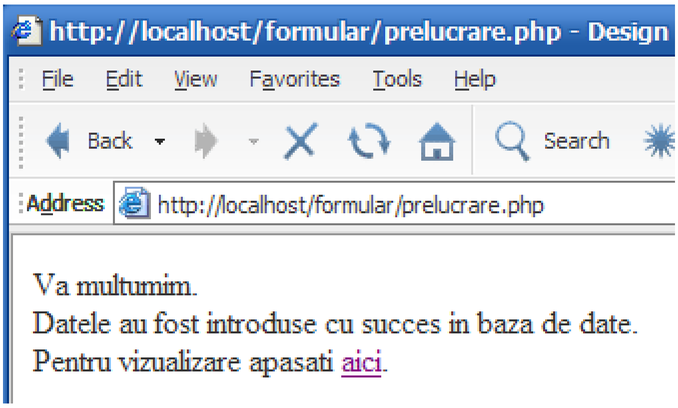
In conditia if, verificam daca s-au introdus date in formular, daca valoarea campului varsta este numerica, daca valoarea campului comentariu este mai mare de 255 caractere.

Constructia if este deschisa si inchisa de parantezele rotunde ( ), iar in interiorul acestora sunt conditiile:

($_SESSION['nume'] == "") = Daca valoarea campului nume este goala…
. . .
(!is_numeric($_SESSION['varsta'])) = Daca valoarea campului varsta nu este numerica... (!is_numeric).
. . .
(strlen($_SESSION['comentariu']) > 255) = Daca valoarea campului comentariu este mai mare de 255 caractere… (strlen = numar de caractere in sir-ul dat).

Daca va intrebati ce inseamna || din interiorul constructiei if, cititi pagina 43 din acest tutorial.

In conditia else, afisam textul, cum ca datele au fost introduse in baza de date, si efectuam inserarea.



In continuare, vom realiza un script prin care afisam in browser datele introduse in baza de date.

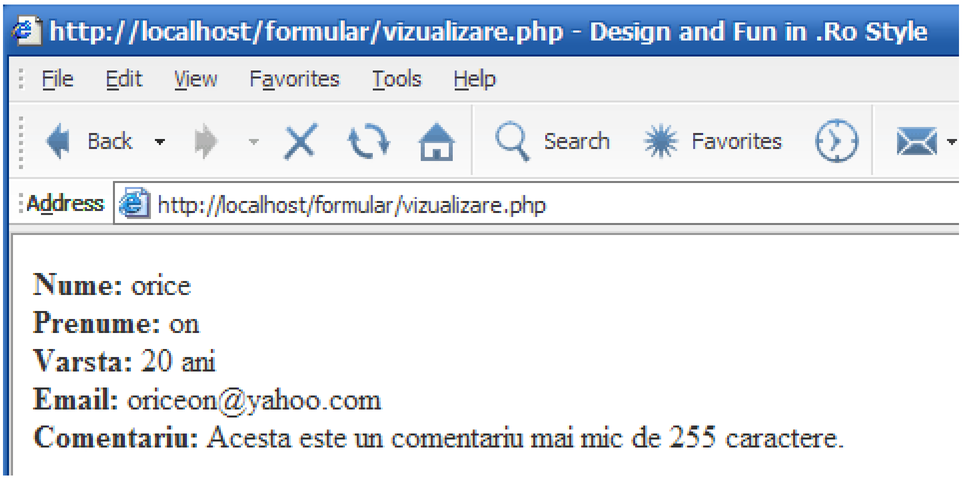
Realizati un fisier cu numele **vizualizare.php** si introduceti codul de mai jos, apoi testati in browser accesand http://localhost/formular/vizualizare.php

```
<?php
require_once('config.php');

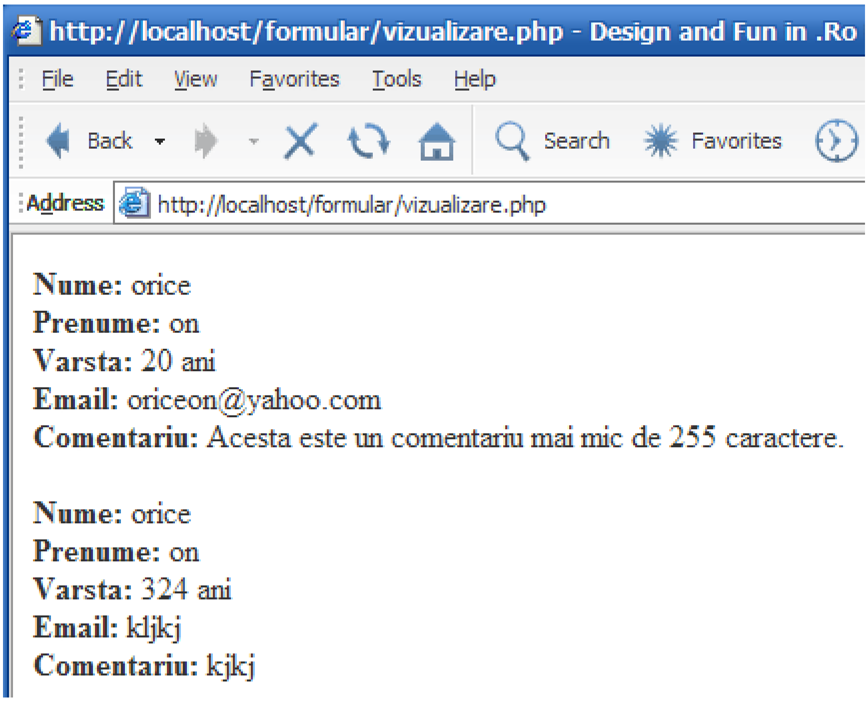
$cerereSQL = 'SELECT * FROM intrari';
$rezultat = mysql_query($cerereSQL);

     while($rand = mysql_fetch_array($rezultat)) {
          echo '<b>Nume:</b> '.$rand['nume'].' <br>
          <b>Prenume:</b> '.$rand['prenume'].' <br>
          <b>Varsta:</b> '.$rand['varsta'].' ani <br>
          <b>Email:</b> '.$rand['email'].' <br>
          <b>Comentariu:</b> '.$rand['comentariu'].' <br><br>';
     }
?>
```

Accesand adresa data, observati ca sunt afisate datele introduse in baza de date:



Introduceti mai multe date completand din nou formularul http://localhost/formular/index.php, apoi accesati http://localhost/formular/vizualizare.php si observati blocurile de date listate unul sub altul.



Recapitulare:

1) Am creat baza de date cu numele formular, tabela intrari si coloanele id | nume | prenume | varsta | email | comentariu cu tipul de date corespunzator pentru fiecare coloana.
2) Am creat fisierul index.php in care am introdus codul HTML cu un formular cu campurile cerute, si anume: nume | prenume | varsta | email | continut si am setat actiunea formularului sa fie catre prelucrare.php (adica <form … action=”validare.php” …>).
3) Am creat fisierul prelucrare.php in care am inclus fisierul config.php cu ajutorul functiei require_once(‘’); In acest script, prelucrare.php, ne-am folosit de constructia if si else pentru a seta conditiile in care scriptul nostru introduce datele in baza de date sau nu.
   Ne-am folosit de cunostintele acumulate in acest tutorial pentru a realiza constructia if in care sa punem conditiile de verificare a campurilor formularului nostru (daca sunt goale, daca campul varsta nu este numeric si daca campul comentariu este mai mare de 255 caractere).
4) Am creat fisierul vizualizare.php in care am inclus fisierul config.php, apoi am efectuat interogarea bazei de date si extragerea coloanelor in functie de dorintele noastre.

Tema

Problema:

Se cer urmatoarele campuri: nume (tipul text), varsta (tipul text), oras (tipul text), ocupatie (tipul text), codul numeric personal (tipul text).

Cerinta:

Realizati baza de date, verificati campurile (sa fie obligatorii, campurile nume si oras trebuie sa contina numai litere, campul cod numeric personal sa contina numai cifre si sa nu poata fie introduse mai mult de 13 caractere), introduceti in baza de date informatiile prelucrate prin formular si afisati-le in ordinea:
nume | oras | ocupatie | varsta | cod numeric personal.

Scriptul complet il puteti descarca de la adresa:
http://www.oriceon.com/tutorial/descarca/exemple/ex1-formular.zip

Exercitiul 2

Realizarea unui sistem de inregistrare, autentificare si protectie a unor pagini

Problema:

Pentru sistemul de inregistrare se cer urmatoarele campuri: utilizator (tipul text), parola (tipul password), reintroducere parola (tipul password), nume (tipul text), prenume (tipul text), varsta (tipul text), localitate (tipul text).

Cerinta:

Toate campurile sa fie obligatorii, valoarea campurilor parola1 si parola2 trebuie sa fie aceeasi, campurile nume, prenume si localitate trebuie sa contina numai litere, campul varsta sa contina numai cifre.

1) Realizati o pagina cu un formular de inregistrare si introduceti datele, dupa validare, in baza de date.
2) Realizati o pagina de autentificare.
3) Restrictionati o pagina pentru utilizatorii care nu sunt autentificati.
4) In pagina utilizatorilor inregistrati, realizati un **panou de control** in care utilizatorul sa isi poata schimba datele personale, parola.

Rezolvare:

Prima si prima oara trebuie sa realizam baza de date.
Intram in phpMyAdmin si cream o baza de date cu numele autentificare apoi tabela utilizatori cu 7 coloane si anume:

id | utilizator | parola | nume | prenume | varsta | localitate

(Dupa cum am invatat mai sus in capitolul MySQL, campul ID trebuie sa fie de tipul INT, auto_increment si primary, apoi campurile: utilizator CHAR(60), parola CHAR(30), nume CHAR(30), prenume CHAR(30), varsta CHAR(10), localitate CHAR(30)).

Table utilizatori has been created.

SQL query:

```
CREATE TABLE `utilizatori` (
  `id` INT NOT NULL AUTO_INCREMENT ,
  `utilizator` CHAR( 60 ) NOT NULL ,
  `parola` CHAR( 60 ) NOT NULL ,
  `nume` CHAR( 30 ) NOT NULL ,
  `prenume` CHAR( 30 ) NOT NULL ,
  `varsta` CHAR( 3 ) NOT NULL ,
  `localitate` CHAR( 30 ) NOT NULL ,
  PRIMARY KEY ( `id` )
) TYPE = MYISAM ;
```

[Edit] [Create PHP Code]

	Field	Type	Collation	Attributes	Null	Default	Extra	Action
☐	id	int(11)			No		auto_increment	
☐	utilizator	char(60)	latin1_swedish_ci		No			
☐	parola	char(60)	latin1_swedish_ci		No			
☐	nume	char(30)	latin1_swedish_ci		No			
☐	prenume	char(30)	latin1_swedish_ci		No			
☐	varsta	char(3)	latin1_swedish_ci		No			
☐	localitate	char(30)	latin1_swedish_ci		No			

Dupa ce ati realizat baza de date, intrati in directorul radacina al serverului dvs (daca folositi EasyPHP este folderul www) si creati un folder cu numele **autentificare**.

Pentru a intelege mai bine scriptul, va voi ruga sa il descarcati de la adresa http://www.oriceon.com/tutorial/descarca/exemple/ex2-autentificare.zip si apoi sa il comentam.

Extrageti arhiva in directorul radacina al serverului dumneavoastra. Observati 7 scripturi PHP.

config.php – Aici gasiti datele de autentificare la baza dumneavoastra de date (modificati cu datele dvs).
index.php – Pagina principala cu legaturi catre scripturile de inregistrare si autentificare useri.
inregistrare.php – script pentru inregistrare de noi utilizatori.
autentificare.php – script pentru autentificare utilizatori.
pagina.php – pagina accesibila numai utilizatorilor autentificati.
profil.php – pagina pentru modificare date personale.
iesire.php – pagina pentru iesire din sistem.

Dupa ce ati modificat datele din **config.php** cu cele de la baza dvs de date, deschideti pagina inregistrare.php si aruncati o privire (mare atentie deoarece in acest exemplu de script o sa invatam multe lucruri noi).

Scriptul de inregistrare

Ca o a II-a linie din script, observam linia if(!isset($_GET['actiune'])) $_GET['actiune'] = '';
Aceasta linie, daca nu gaseste $_GET['actiune'] ca fiind setata, o va seta.

La fel si restul:

```
if(!isset($_SESSION['user'])) $_SESSION['user'] = '';
if(!isset($_SESSION['parola1'])) $_SESSION['parola1'] = '';
if(!isset($_SESSION['parola2'])) $_SESSION['parola2'] = '';
if(!isset($_SESSION['nume'])) $_SESSION['nume'] = '';
if(!isset($_SESSION['prenume'])) $_SESSION['prenume'] = '';
if(!isset($_SESSION['varsta'])) $_SESSION['varsta'] = '';
if(!isset($_SESSION['localitate'])) $_SESSION['localitate'] = '';
```

Observam constructia if, care are ca si conditie negativa (**!**) o functie isset (se foloseste numai pentru a verifica daca o variabila exista, returneaza adevarat atunci cand o gaseste si fals atunci cand nu o gaseste).
Prin urmare, cu aceasta constructie if.. verificam daca $_GET['actiune'] este setat, daca nu .. o setam noi.

In continuare, uitandu-ne peste script, observam constructia **switch**. Am folosit aceasta pentru a implementa mai multe actiuni intr-o singura pagina (vezi pagina 19).

Dupa cum vedem, primul case este null.. ceea ce inseamna ca se executa scriptul pentru pagina curenta .php

Avem un formular in care sunt 7 campuri: utilizator, parola, reintroducerea parolei, nume, prenume, varsta si localitate.

Daca va uitati atenti, o sa observati ca in valoarea campurilor avem declarate niste sesiuni:
<input type="text" name="user" value="**'.$_SESSION['user'].'**">

Acum, va intrebati ceea ce sunt sesiunile si cum se folosesc ele; in continuare am sa va dau cateva exemple pentru a intelege lucrul cu sesiuni.

Scopul unei sesiuni este de a retine o informatie care sa se mentina de la o pagina la alta.
Pentru a seta o sesiune si a o mentine de la o pagina la alta, este nevoie de a o declara cu ajutorul unei functii.. si anume: **session_start();**
Aceasta functie se pune la inceput de PAGINA, prima linie din script.

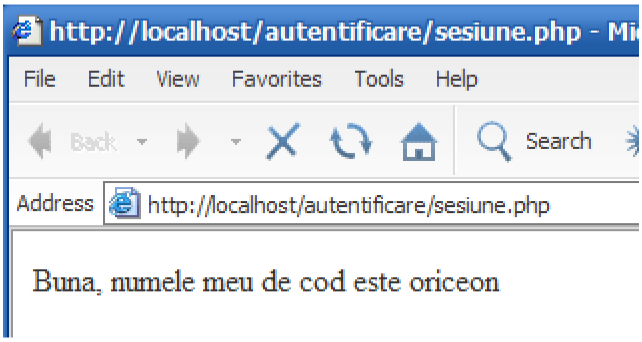
Constructia unei sesiuni este: $_SESSION['nume_sesiune'] = 'valoare sesiune';

Exemplu:

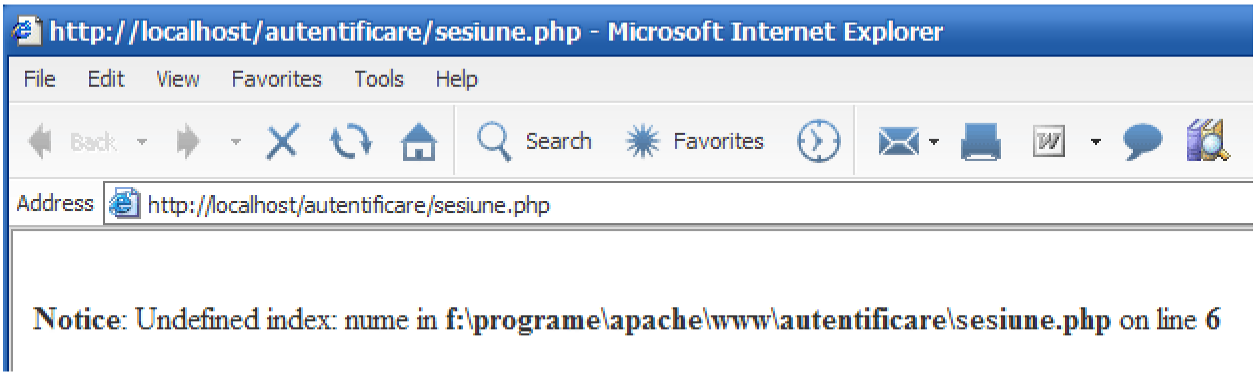
```
<?php
session_start();

$_SESSION['nume'] = 'Buna, numele meu de cod este oriceon';
echo $_SESSION['nume'];
?>
```

Puneti acest cod intr-un fisier si apoi denumiti-l sesiune.php, apoi testati in browser.



Daca o sa inlaturati prima linie din cod, si anume declararea sesiunii cu session_start(); veti observa urmatoarea eroare.



Concluzie: nu uitat sa setati ca prima linie de cod, session_start();

Pentru a distruge sesiunile, ne putem folosi de o functie care va sterge toate sesiunile existente.. si anume **session_destory();** sau de declararea sesiunilor ca fiind nule: $_SESSION['nume_sesiune'] = '';

Intorcandu-ne la formularul nostru de inregistrare, ati vazut ca valoarea campurilor .. este o sesiune.
Vom seta cate o sesiune pentru fiecare camp, astfel incat sa pastram datele scrise de el, iar daca utilizatorul completeaza gresit campurile si este redirectionat la pagina cu formularul.. datele introduse de el vor aparea din nou in formular, astfel incat el va putea sa corecteze campurile in care a gresit.

Ca actiune a formularului este setat inregistrare.php?actiune=validare
Acest case il gasiti la linia 112.

Dupa cum vedeti, avem declarate sesiunile de care ne vom folosi in valoarea campurilor din formular.. si anume:

$_SESSION['user'] = $_POST['user'];

Valoarea acestei sesiuni user, este valoarea POST cu numele user… si tot asa.

In urmatoarele linii de cod avem structura if si else cu conditiile necesare, iar daca conditiile nu sunt adevarate, ne rezulta eroarea, daca sunt acceptate.. rezulta mesajul de multumire si inserarea in baza de date.. apoi setarea sesiunilor folosite ca fiind nule.

Dupa acest case cu numele verifica.. vom inchide structura switch, respectiv scriptul php.

Observam ca la introducerea in baza de date am complicat procedeul.
Motivul este pentru a ne proteja de procedeul numit SQL INJECTION (vezi capitolul de securitate pagina 79)
Aceasta operatiune se foloseste pentru a introduce vulnerabilitati in comanda care trimite datele la baza de date (comanda query) si cu ajutorul acestora, se poate pacali cu usurinta baza de date.

Pentru a proteja datele care se introduc prin formular, ne-am folosit de o functie, si anume: addentities();

Exemplu de script de atac:

```
SELECT * FROM `tabela` WHERE camp = 'orice' OR 'x'='x';
```

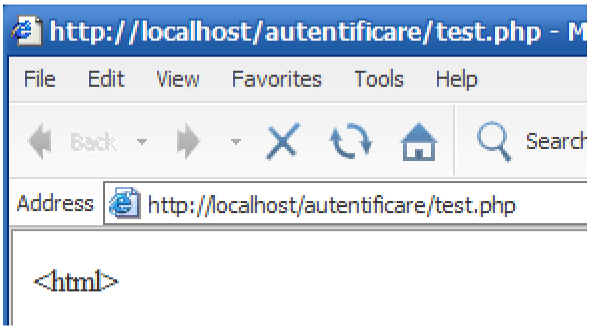
Observati ca utilizatorul poate pacali comanda SQL, introducand ca VALOARE a campului.. <u>orice' OR 'x'='x</u> astfel incat comanda SQL se va “modifica”.

Functia addlentities(); va transforma caracterele speciale din html in coduri.

Exemplu: din codul html <html> ne va rezulta <html> sau daca utilizatorul introduce ‘ sau “ in campul innput, acestea se vor transforma in: ' sau " . . .

Protectia cu aceasta functie se face pentru ca atunci cand citim din baza de date un cod html, sa nu fie interpretat de catre browser ci.. doar sa fie afisat.

Puneti intr-un fisier test.php codul: <html> si apoi puneti si <html> .. testati si observati rezultatul.



Urmatoarea noutate din comanda noastra SQL este functia md5();
Aceasta functie am folosit-o pentru a coda parola noastra si astfel a o introduce in baza de date.. neputand fi decodata de catre nimeni.

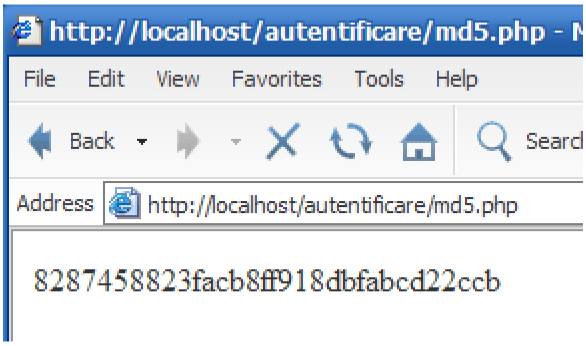
md5 este o functie PHP care codeaza o parola in mod teoretic ireversibil..

```php
<?php

$parola = 'parola';
echo md5($parola);

?>
```

Puneti acest cod intr-un fisier md5.php apoi testati si veti observa urmatorul rezultat:



Pana acum am tot vorbit de functii, functii si iarasi functii si presupun ca va intrebati ce sunt acestea.

O functie este folosita pentru a separa codul care realizeaza un singur task bine definit. Acest lucru face codul mai lizibil si ne permite sa il reutilizam de fiecare data cand trebuie sa efectuam acelasi task.

Argumentele unei functii trebuie separate prin virgula, si, implicit, acestea sunt transmise prin valoare. Pentru ca functia sa returneze un rezultat se foloseste constructia **return** care primeste ca parametru o expresie care reprezinta valoarea functiei. In momentul in care este intalnita constructia **return**, executia functiei se incheie.

In PHP sunt foarte multe functii predefinite, insa puteti sa va concepeti si propriile functii.
Apelul unei functii se face: nume_functie();

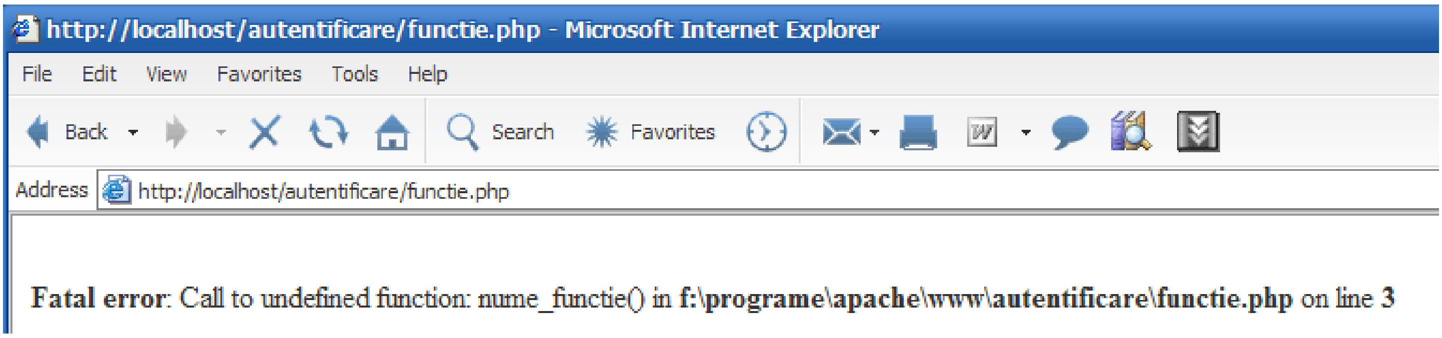
Majoritatea functiilor necesita unul sau mai multi parametrii – informatii oferite functiei atunci cand este apelata si care influenteaza rezultatul executiei functiei –
Apelul unei functii cu parametru se realizeaza astfel: nume_functie($parametru);

Daca incercati sa apelati o functie care nu exista, veti primi un mesaj de eroare.

```
<?php

nume_functie();

?>
```



Realizarea primei noastre functii:

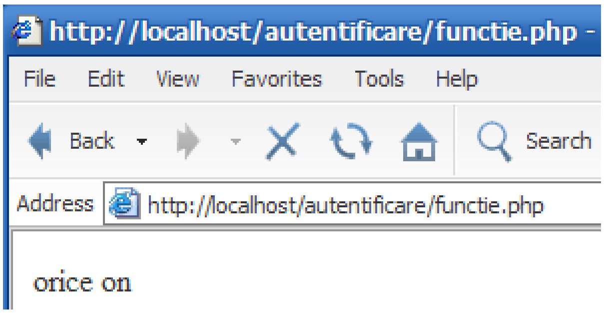
```
<?php

function functie_nume($nume) {
$prenume = 'on';
return $nume.' '.$prenume;
}

echo functie_nume('orice');

?>
```

Realizati un fisier cu numele functie.php apoi testati in browser.



Exemplul dat este unul simplu, functiile se folosesc pentru lucruri mult mult mai dificile, insa sper ca v-ati facut o idee asupra functiilor.

In exemplul urmator se calculeaza cu ajutorul unei functii PHP, patratul unui numar.

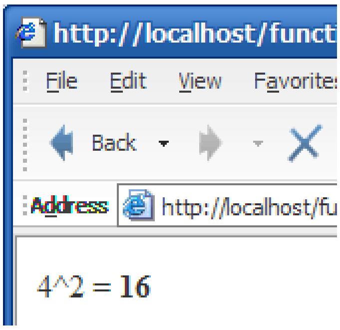
```
<?php

function patrat($n) {
    return $n * $n;
}

echo '4^2 = <b> '.patrat(4).' </b>';

?>
```

Observati ca am atribuit valoarea 4 functiei noastre cu numele patrat, iar rezultatul apelarii acesteia este calculul din interiorul functiei: `return $n * $n;` adica returneaza 4 x 4, care ne da 16.



Revenind la scriptul nostru de inregistrare, testati-l, apoi verificati daca datele au fost introduse in baza de date.

Scriptul de autentificare

La fel ca si la scriptul de inregistrare, observam setarea $_GET['actiune'] ca fiind nula, apoi deschiderea structurii switch.

In case null avem formularul de autentificare ce cuprinde 2 campuri: utilizator si parola.

In case validare, avem sesiunea user cu valoarea postului user urmata de constructia if si else cu conditiile aferente.

Sa va invat un truc.. pentru a vedea de unde si pana unde tine o constructie if.
Apasati un click in fata unei acolade { si apoi veti observa ca o sa se inroseasca.

Exemplu:

```
20  if(($_POST['user'] == '') || ($_POST['parola'] == ''))
21  {
22  echo 'Completeaza casutele. <Br>
23        Apasati <a href="autentificare.php">aici</a> pentru a va intoarce la pagina precedenta.';
24  }
25  else
26  {
```

Observati ca apare acolada unde se inchide if-ul nostru. (liniile 21 si 24).

Apasati apoi si in fata acoladei de la else si observati pana unde tine acea conditie (observam ca tine de la linia 26 pana la linia 43).

In aceasta constructie else, avem selectarea din baza de date unde campul utilizator este egal cu POST user, iar parola este egala cu criptarea parolei pentru POST parola.

Daca numarul de randuri rezultate este 1, atunci setam o sesiune cu numele logat si valoarea Da, apoi redirectionam - cu ajutorul etichetei HTML meta refresh – catre pagina utilizatorului.

Daca numarul de randuri rezultate nu este 1, atunci afisam un mesaj de eroare.

In cazul in care rezultatele au fost gasite in baza de date, autentificarea s-a efectuat cu succes si vom fi redirectionati in pagina.php

Scriptul pagina utilizator

Observam ca avel o constructie if si else.

Daca sesiunea cu numele logat nu are valoarea Da executam o bucla de cod, daca sesiunea logat are valoarea Da.. executam pagina pentru utilizator.

Sa ne reamintim ca aceasta sesiune $_SESSION['logat'] == 'Da'; a fost setata atunci cand autentificarea s-a realizat cu succes.

Accesati http://localhost/autentificare/pagina.php, fara sa va autentificati, si veti observa mesajul de eroare, apoi autentificati-va si veti observa pagina utilizatorului.

Din aceasta pagina, va puteti schimba datele personale precum si parola, iar apoi puteti sa iesiti din sistem folosind link-ul "iesire".

Scriptul profil

Si in acest script, la fel ca si in celelalte, ne-am folosit de constructia switch si case.

In case null, avem legaturile catre urmatoarele case unde vom avea formularele de prelucrare a datelor existente utilozatorului autentificat.

In case 'date_personale': selectam tot din baza de date unde utilizator este egal cu utilizatorul autentificat. Daca conexiunea returneaza rezultate, executam formularul de prelucrare a datelor, iar in campul value al fiecarui input vom seta ca valoare campul din baza de date aferent acelui input.

Exemplu:

```
<input type="text" name="nume" value="'.$rand['nume'].'">
```

Observati ca acest case, precum si cel de modificare parola, se foloseste de aceeasi actiune, respectiv:
profil.php?actiune=validare

Da, este posibil deoarece am setat 2 valori diferite pentru cele 2 formulare:
Pentru formularul cu date am setat ca valoare: “Modifica date” iar pentru cel cu parola am setat “Modifica parola”, apoi, in functie de ce buton era apasat, am realizat conditiile if.

Scriptul iesire

Acest script contine functiile ce goleste si distruge sesiunile.

Recapitulare:

1) Am creat baza de date cu numele autentificare, tabela utilizatori si coloanele id | utilizator | parola | nume | prenume | varsta | localitate cu tipul de date corespunzator pentru fiecare coloana.
2) Am realizat fisierul de configurare (config.php) unde am setat conexiunea la baza de date.
3) Am realizat scriptul de inregistrare si cel de autentificare, scripturi in care am invatat ce sunt si cum sa folosim sesiunile, functiile, si am aprofundat SQL INJECTION folosindu-ne de functiile addslashes si htmlentities precum si de criptarea parolelor cu functia md5.
4) Am realizat scriptul pagina.php, script accesibil numai persoanelor autentificate. Din aceasta pagina am realizat modalitatea de a schimba datele personale, precum si parola utilizatorului autentificat.
5) Ultima pagina din script este cea in care distrugem sesiunile si iesim din sistem.

Tema

Modificati acest sistem, astfel incat orice utilizator sa poata introduce si sa citeasca un text in baza de date.

Sfat! Realizati inca o casuta in baza de date pe langa cele existente si apoi realizati o pagina cu un formular si un cam textarea, verificati datele introduse prin formular, apoi inserati in baza de date unde utilizator = sesiunea user.

Scriptul complet il puteti descarca de la adresa:
http://www.oriceon.com/tutorial/descarca/exemple/ex2-autentificare.zip

Exercitiul 3

Realizarea unui formular de contact

Problema:

Realizati un formular de contact cu 5 campuri: E-Mail, Nume, Prenume, Subiect, Mesaj si, dupa validare, trimiteti datele intr-un email folosindu-va de functia mail();

Cerinta:

Toate campurile sunt obligatorii, campurile nume, prenume sa contina numai litere.

1) Realizati o pagina cu un formular ce sa contina campurile de mai sus.
2) Realizati o pagina de prelucrare si trimitere a mesajului.

Rezolvare:

Creati un folder cu numele **contact** in directorul www apoi creati un fisier cu numele **index.php** si introduceti codul urmator:

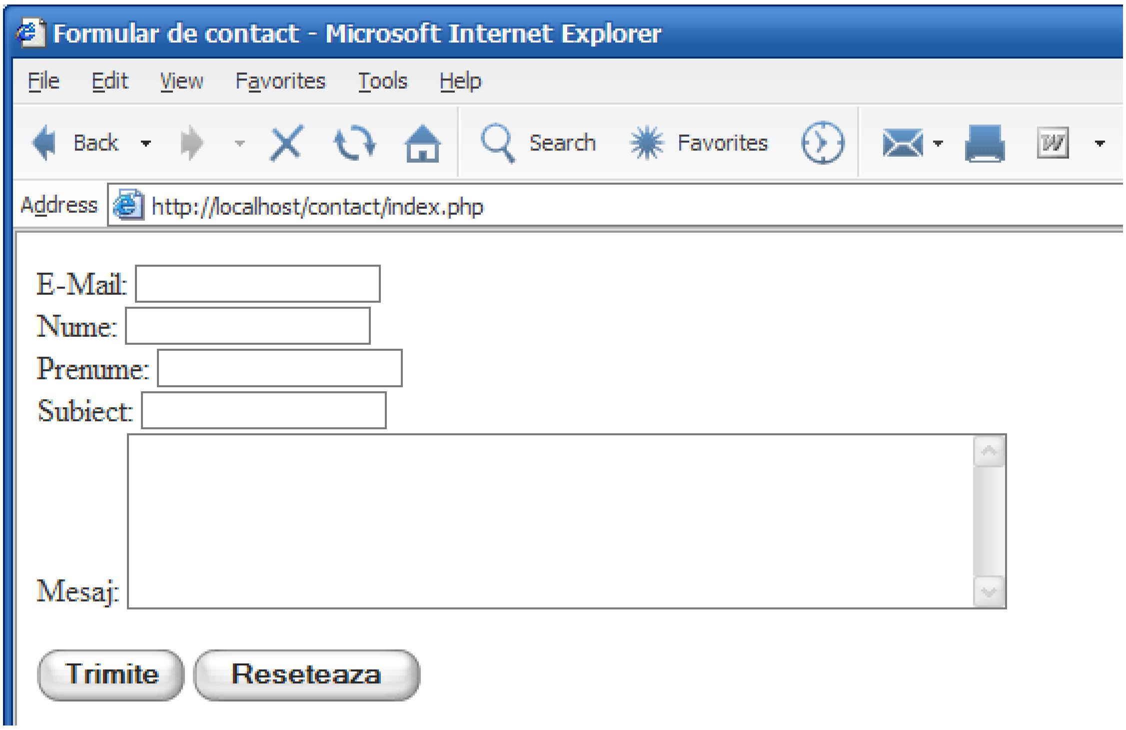
```
<html>
<head>
<title>Formular de contact</title>
</head>
<body>
<form action="trimite.php" method="post">

E-Mail:  <input type="text" name="email" value=""> <br>
Nume:    <input type="text" name="nume" value=""> <br>
Prenume: <input type="text" name="prenume" value=""> <br>
Subiect: <input type="text" name="subiect" value=""> <br>
Mesaj:   <textarea name="mesaj" cols="60" rows="6"></textarea> <br><br>

<input type="submit" name="Trimite" value="Trimite">
<input type="reset" name="Reseteaza" value="Reseteaza">

</form>
</body>
</html>
```

Salvati si vizualizati in browser accesand http://localhost/contact/index.php



Creati un fisier cu numele **trimite.php** si introduceti codul de mai jos:

```
<?php

if(($_POST['email'] == '') || ($_POST['nume'] == '') || (is_numeric($_POST['nume'])) ||
($_POST['prenume'] == '') || (is_numeric($_POST['prenume'])) || ($_POST['subiect'] == '')
|| ($_POST['mesaj'] == '')) {
echo 'Completati campurile corect <br>
      Apasati <a href="index.php">aici</a> pentru a va intoarce la pagina principala.';
} else {

$catre = 'oriceon@yahoo.com';
$data_trimitere = date('d-m-Y H:i:s');

$subiect = $_POST['subiect'];

$mesaj = '
<html>
<head>
<title>Formular de Contact</title>
</head>
<body>
<p><tt>Data trimitere: '.$data_trimitere.' </tt></p>
<table>
<tr>
<td><tt> Nume: '.$_POST['nume'].' </tt></td>
</tr>
<tr>
<td><tt> Prenume: '.$_POST['prenume'].' </tt></td>
</tr>
<tr>
<td><tt> E-Mail: <a href="mailto:'.$_POST['email'].'">'.$_POST['email'].'</a> </tt></td>
</tr>
<tr>
<td><tt> Mesaj: <br><br> '.$_POST['mesaj'].' </tt></td>
</tr>
</table>
</body>
</html>';
```

```
$headere  = "MIME-Version: 1.0\r\n";
$headere .= "Content-type: text/html; charset=iso-8859-1\r\n";
headere .= "From: ".$_POST['nume']." ".$_POST['prenume']."<".$_POST['email'].">\r\n";

mail($catre, $subiect, $mesaj, $headere);

echo 'Mesajul a fost trimis';

}

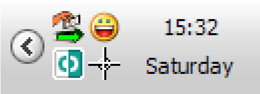
?>
```

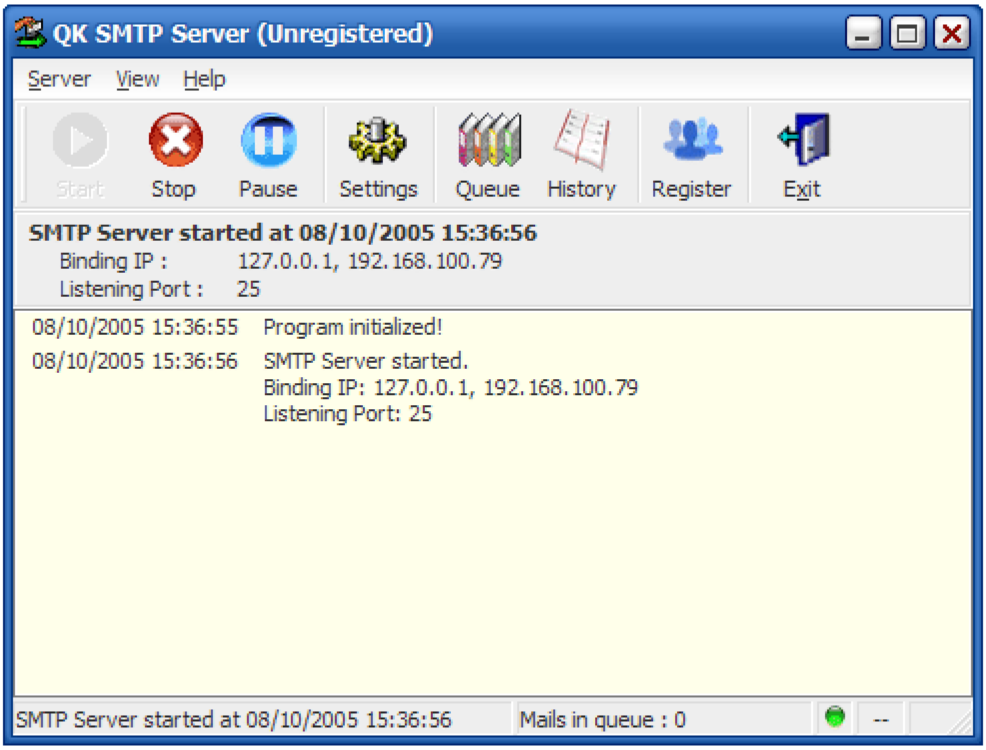
Modificati adresa de email din variabila `$catre = 'oriceon@yahoo.com';` cu email-ul dvs.

Pentru a face acest script sa trimita mail, aveti nevoie de un server SMTP, un server de E-Mail.

Va recomand serverul **QK** pe care il puteti descarca de la adresa http://www.qksoft.com/

Dupa ce ati instalat serverul si l-ati pornit, ar trebui sa va apara in bara de start de langa ceas.





Testati scriptul si verificati email-ul dvs pentru a vedea daca ati primit mesajul. (In caz ca nu apare in Inbox, uitati-va in Bulk).

Observati ca in scriptul **trimite.php** ne-am folosit de conditiile if si else (conditii pe care deja le stiam) insa am folosit si 2 noi functii, si anume: date(); si mail();

Functia date();

Aceasta functie returneaza un string ce contine data.
Acestei functii i se pot atribui parametrii, parametrii ce semnifica formatul de afisare a datei.

Parametrii pentru functia date();

Parametru	Descriere	Exemplu
a	Returneaza Ante meridian sau Post meridian in caractere mici	am sau pm
A	Returneaza Ante meridian sau Post meridian in caractere mari	AM sau PM
d	Ziua din luna, 2 caractere (cu 0)	01 pana la 31
D	Reprezentarea text a zilei, trei litere (in engleza)	Mon pana la Sun
F	Reprezentarea text a lunii	January pana la December
g	Formatul a 12 ore (fara 0)	1 pana la 12
G	Formatul a 24 ore (fara 0)	0 pana la 23
h	Formatul a 12 ore ce (cu 0)	01 pana la 12
H	Formatul a 24 ore ce (cu 0)	01 pana la 23
i	Minute ce (cu 0)	00 pana la 59
j	Ziua din luna (fara 0)	1 pana la 31
l (caracter mic L)	Reprezentarea text a zilei din saptamana	Sunday pana la Saturday
m	Reprezentarea numerica a lunii (cu 0)	01 pana la 12
M	Reprezentarea text a lunii, trei litere	Jan pana la Dec
n	Reprezentarea numerica a lunii (fara 0)	1 pana la 12
s	Secunde (cu 0)	00 pana la 59
t	Numarul de zile din luna data	28 pana la 31
w	Reprezentarea numerica a zilei din saptamana	0 (pentru Sunday) pana la 6 (pentru Saturday)
y	Reprezentarea numerica a anului (2 cifre)	Exemplu: 05
Y	Reprezentarea numerica a anului (4 cifre)	Exemplu: 2005
z	Ziua din an	0 pana la 365

Aceasta functie returneaza datele in engleza, precum vedeti la F, l, M, pentru a returna in romana, trebuie sa va faceti propria functie, insa, pana atunci, o sa lucram cu functia predefinita, date();

Exemple de folosire a functiei date();

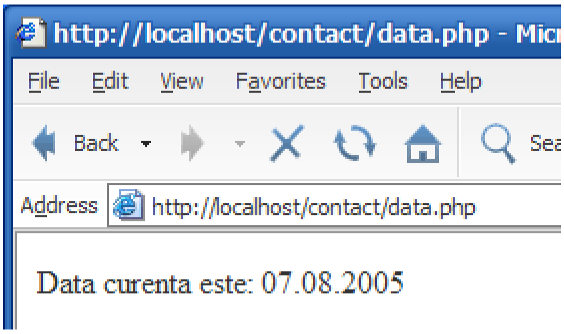
Realizati un fisier cu numele **data.php** si introduceti codul de mai jos:

```
<?php

$data = date("d.m.Y");
echo 'Data curenta este: '.$data.'';

?>
```

Salvati si vizualizati in browser http://localhost/contact/data.php

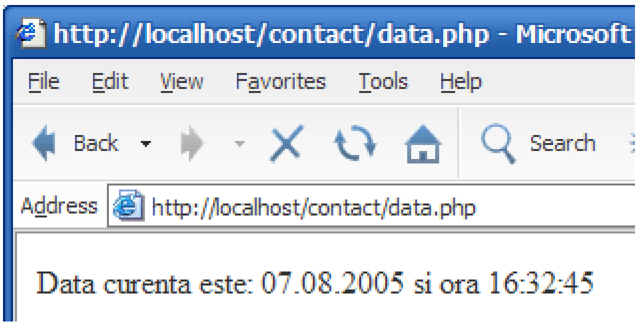


Si asa mai departe va puteti "juca" cu functia date si parametrii ei pentru a obtine rezultatele dorite:

```
<?php

$data = date("d.m.Y");
$ora = date("H:i:s");
echo 'Data curenta este: '.$data.' si ora '.$ora.'';

?>
```



Functia mail();

Dupa cum vedeti in scriptul trimite.php, ne-am folosit de aceasta functie PHP pentru a trimite datele culese din formular in email-ul nostru.

Un simplu exemplu de folosire ar fi urmatorul:

```
<?php
mail("oriceon@yahoo.com", "Subiectul meu", "Line 1\nLine 2\nLine 3");
?>
```

Pentru ca trimiterea mail-ului sa se faca mai rapid si mai usor si sa fie mai usor de interpretat de catre serverul de email si client, trebuie sa ii definim parametrii pentru headere.

```
<?php
mail("oriceon@yahoo.com", "Subiectul", "Mesajul",
     "From: oriceon@gmail.com\r\n" . "Reply-To: oriceon@hotmail.com\r\n"."X-Mailer: PHP/"
. phpversion());
?>
```

Acum, analizand exemplul nostru de trimitere email din scriptul trimite.php, observam ca de fapt nu este asa greu precum pare, ci chiar este foarte foarte usor:

Dupa cum vedeti, am declarat niste variabile:

```
$catre = 'oriceon@yahoo.com';    // adresa de email unde primim informatiile
$data_trimitere = date('d-m-Y H:i:s'); // data la care mesajul a fost trimis

$subiect = $_POST['subiect']; // subiectul (pe care l-a completat utilizatorul in form)
```

Apoi avem variabila `$mesaj` ce contine ca valoare un cod HTML cu datele trimise prin formular si variabila `$data_trimitere.`

Si, ca ultima alcatuire, avem headerele si functia mail ce contine variabilele setate mai sus.

```
mail($catre, $subiect, $mesaj, $headere);
```

Recapitulare:

1) Am creat un formular cu 5 camprui: E-Mail, Nume, Prenume, Subiect si Mesaj
2) Am creat un script de prelucrare a datelor si ne-am folosit de conditiile if si else, apoi, in conditia else, am folosit doua noi functii: **date()** si **mail()**

Tema

Prin prisma cunostintelor acumulate din exemplele anterioare, realizati o baza de date in care sa aveti 2 tabele: administrator si mesaje. Realizati un formular prin care utilizatorul sa va contacteze (ca cel de mai sus) si apoi, in conditia else, realizati inserarea in baza de date, precum si trimiterea unui email catre dvs pentru a va anunta de un nou mesaj de contact.

Accesati apoi sectiunea de administrare si vizualizati mesajele introduse in baza de date direct din browser. Pentru aceasta, trebuie sa realizati o pagina php prin care sa selectati * din tabela mesaje si sa le listati.

Scriptul de contact folosit in tutorial il puteti descarca de la adresa:
http://www.oriceon.com/tutorial/descarca/exemple/ex3-contact.zip

Exercitiul 4

Realizarea unui sistem de votare

Problema:

Realizati un formular cu intrebarile luate din baza de date si cate un input de tip radio in fata fiecarei intrebari. Creati o baza de date cu numele **votare** si cu 2 tabele **intrebare** si **optiuni**.

Tabela intrebare trebuie sa aibe coloanele id (INT) si intrebare (VARCHAR(255)).

Table intrebare has been created.

SQL query:

```
CREATE TABLE `intrebare` (
  `id` INT NOT NULL AUTO_INCREMENT ,
  `intrebare` VARCHAR( 255 ) NOT NULL ,
  PRIMARY KEY ( `id` )
) TYPE = MYISAM ;
```

[Edit] [Create PHP Code]

	Field	Type	Collation	Attributes	Null	Default	Extra	Action
☐	id	int(11)			No		auto_increment	
☐	intrebare	varchar(255)	latin1_swedish_ci		No			

Iar tabela optiuni, coloanele, id (INT), optiune (VARCHAR(255)), voturi(BIGINT(30)).

Table optiuni has been created.

SQL query:

```
CREATE TABLE `optiuni` (
  `id` INT NOT NULL AUTO_INCREMENT ,
  `optiune` VARCHAR( 255 ) NOT NULL ,
  `voturi` BIGINT( 30 ) NOT NULL ,
  PRIMARY KEY ( `id` )
) TYPE = MYISAM ;
```

[Edit] [Create PHP Code]

	Field	Type	Collation	Attributes	Null	Default	Extra	Action
☐	id	int(11)			No		auto_increment	
☐	optiune	varchar(255)	latin1_swedish_ci		No			
☐	voturi	bigint(30)			No	0		

Cerinta:

1) Realizati o pagina unde sa listati intrebarea si optiunile din baza de date si in fata fiecarei optiuni puneti un input type radio.
2) Realizati o pagina de prelucrare a optiunii alese si updatarea bazei de date cu + 1

Rezolvare:

Creati un folder cu numele **votare** in directorul www, creati un fisier de configurare cu numele **config.php** si introduceti codul urmator:

```
<?php
session_start();
set_time_limit(0);
error_reporting(E_ALL);

// Informatii baza de date

    $AdresaBazaDate = "localhost";
    $UtilizatorBazaDate = "root";
    $ParolaBazaDate = "parola_baza";
    $NumeBazaDate = "votare";

$conexiune = mysql_connect($AdresaBazaDate,$UtilizatorBazaDate,$ParolaBazaDate)
or die("Nu ma pot conecta la MySQL!");
mysql_select_db($NumeBazaDate,$conexiune) or die("Nu gasesc baza de date!");

function addentities($data){
    if(trim($data) != ''){
    $data = htmlentities($data, ENT_QUOTES);
    return str_replace('\\', '&#92;', $data);
    } else return $data;
} // End addentities() --------------

?>
```

Apoi creati un fisier **index.php**:

```
<?php
require_once('config.php');

$cerereSQL = 'SELECT * FROM `intrebare`';
$rezultat = mysql_query($cerereSQL);
while($rand = mysql_fetch_array($rezultat)) {
echo '<table width="294" border="0" cellpadding="0" cellspacing="0">
<form name="votare" action="voteaza.php" method="post">
<tr>
  <td width="294" height="28" valign="top"><p>'.$rand['intrebare'].'</p></td>
    </tr>';
}

$cerereSQL = 'SELECT * FROM `optiuni`';
$rezultat = mysql_query($cerereSQL);
while($rand = mysql_fetch_array($rezultat)) {p
  echo '<tr>
      <td height="19" valign="top">
        <input name="optiune" type="radio" value="'.$rand['id'].'&'.$rand['voturi'].'">
'.$rand['optiune'].' </td>
    </tr>';
}

echo '<tr>
     <td height="14"></td>
     </tr>
   <tr>
     <td height="24" valign="top"><input name="Voteaza" type="submit" id="Voteaza"
value="Voteaza"> <br><br>
  <a href="rezultate.php">Vezi rezultate</a></td>
```

```
    </tr>
  </form>
 </table>';

?>
```

Dupa cum vedeti, selectam intrebarea din tabela “intrebare” si o afisam sus de tot, apoi facem un nou select unde afisam optiunile pentru intrebare, precum si un input type radio in fata fiecarei optiuni.
Ca value avem `value="'.$rand['id'].'&'.$rand['voturi'].'"` adica id-ul si numarul de voturi pentru fiecare optiune. (se ia automat pentru fiecare deoarece este in bucla while).
Realizati apoi un nou fisier cu numele **voteaza.php**, fisier care este setat ca actiune pentru formularul nostru din index.php.

```
<?php
require_once('config.php');

if(!isset($_POST['Voteaza'], $_POST['optiune'])) {
echo 'Pentru a vota trebuie sa selectati o optiune. <br>
      Apasati <a href="index.php">aici</a> pentru a va intoarce.';
} else {

$informatii = explode('&', $_POST['optiune']);
$id = $informatii[0];
$voturi = $informatii[1];

$cerereSQL = "UPDATE `optiuni` SET `voturi` = (".$voturi."+1) WHERE `id` = ".$id."";
mysql_query($cerereSQL);

echo 'Apasati <a href="rezultate.php">aici</a> pentru a vizualiza rezultatele';

}

?>
```

In aceasta pagina observati ca ne folosim de conditiile if si else.
In conditia if avem ca regula functia **isset();**
Numele acestei functii ne sugereaza de fapt ceea ce vrea sa insemne/execute, adica, isset = daca e setat.

Observati ca noi avem negata aceasta functie si anume: !isset(); ceea ce inseamna cam asa:

```
if(!isset($_POST['Voteaza'], $_POST['optiune'])) {
```

Daca butonul “voteaza” nu este setat, sau daca nu este selectata nici o optiune, atunci… {

In interiorul conditiei else, observam o functie noua, si anume functia **explode();**

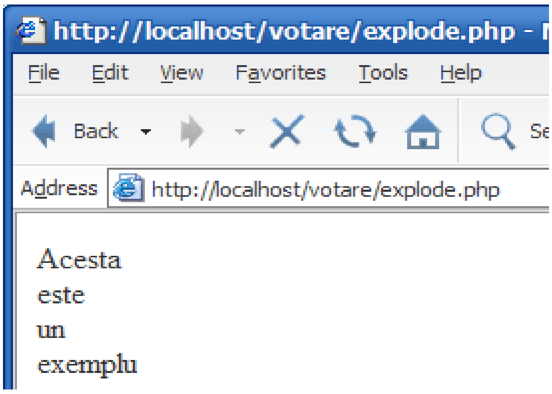
Aceasta functie returneaza un array dintr-un string. Sa luam urmatorul exemplu si sa realizam un fisier cu numele **explode.php** si sa verificam.

```
<?php

$text = 'Acesta este un exemplu de explode';
$cuvinte = explode(' ', $text);

echo $cuvinte[0].'<br>';
echo $cuvinte[1].'<br>';
echo $cuvinte[2].'<br>';
echo $cuvinte[3].'<br>';

?>
```



Observati ca avem o variabila cu numele $text, iar ca valoare a variabilei $cuvinte, avem functia explode();

`explode(' ', $text);` Aceasta functie desparte cuvintele de spatii adica `' '`

Dupa cum vedeti, valoarea variabilei $text contine un text cu spatii intre cuvinte, prin urmare am separat cuvintele scotand spatiile si punandu-le intr-un array.

In continuare vom tipari pe pagina cuvintele rezultate.

```
echo $cuvinte[0].'<br>';
echo $cuvinte[1].'<br>';
echo $cuvinte[2].'<br>';
echo $cuvinte[3].'<br>';
```

Dupa cum vedeti, primul cuvant are numarul 0 iar ultimul 6 (deoarece sunt 6 cuvinte).

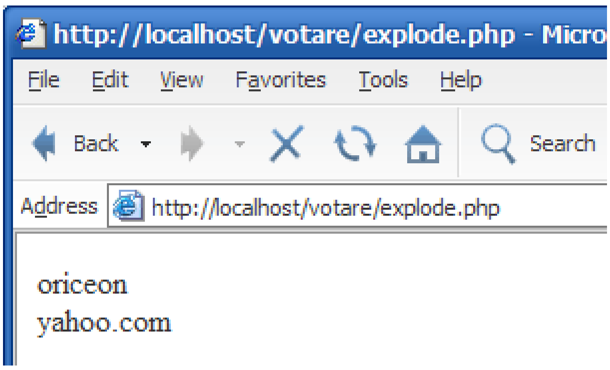
Un alt exemplu:

```
<?php

$email = 'oriceon@yahoo.com';
$cuvinte = explode('@', $email);

echo $cuvinte[0].'<br>';
echo $cuvinte[1].'<br>';

?>
```



Intorcandu-ne la exercitiul nostru, ne aducem aminte ca in scriptul index.php ca valoare a input-ului de tipul radio am avut: `value="'.$rand['id'].'&'.$rand['voturi'].'"`
Am ales aceasta optiune pentru a transfera pe pagina urmatoare valorile ID si VOTURI si pentru a ne folosi de acestea in a updata tabela unde id = id si voturile luate + 1.

```
$informatii = explode('&', $_POST['optiune']);
```

Deci, acest explode, ne separa textul de &.

In continuare, creati un fisier cu numele **rezultate.php** unde vom vizualiza rezultatele votarilor de pana acum.

```
<?php
require_once('config.php');

$cerereSQL = 'SELECT * FROM `intrebare`';
$rezultat = mysql_query($cerereSQL);
while($rand = mysql_fetch_array($rezultat)) {
 echo '<table width="537" border="0" cellpadding="0" cellspacing="0">
   <tr>
     <td height="28" colspan="3" valign="top"><p>'.$rand['intrebare'].'</p></td>
   </tr>';
}

$cerereSQL = 'SELECT * FROM `optiuni`';
$rezultat = mysql_query($cerereSQL);
$totale = mysql_result(mysql_query('SELECT SUM(voturi) FROM `optiuni`'),0);
while($rand = mysql_fetch_array($rezultat)) {

$procent = ($rand['voturi']*100)/$totale;
if($procent <= 0) $procent = 1;

echo '<tr>
     <td width="113" height="19" valign="top"> '.$rand['optiune'].' </td>
     <td width="115" valign="top">'.$rand['voturi'].' voturi </td>
     <td width="309" valign="middle"><img src="linie.gif" width="'.$procent.'%"
height="9"></td>
    </tr>';
}

echo '<tr>
     <td height="14"></td>
     <td></td>
     <td></td>
   </tr>
   <tr>
  <td height="24" colspan="3" valign="top"><a href="index.php">Inapoi la sistemul de
votare</a> </td>
   </tr>
 </table>';

?>
```

Observati ca sus afisam intrebarea, la fel ca si in fisierul index mai sus explicat,

```
$totale = mysql_result(mysql_query('SELECT SUM(voturi) FROM `optiuni`'),0);
```

Aceasta variabila contine ca valoare un SELECT SUM(), ceea ce inseamna un calcul al tutoror intrarilor VOT din baza noastra de date.

Aruncand o privire asupra tabelei optiuni, observam coloanele voturi:

←T→	id	optiune	voturi
	1	Da	22
	2	Nu	23

Folosind select-ul de mai sus, ca rezultat vom avea 22+23 adica 45

In continuare, ne vom folosi de niste calcule matematice, calcule care m-au stresat putin ☺

```
$procent = ($rand['voturi']*100)/$totale;
if($procent <= 0) $procent = 1;
```

Dupa cum vedeti, pentru fiecare optiune luata prin bucla WHILE, avem cate o imagine

```
<img src="linie.gif" width="'.$procent.'%" height="9">
```

Aceasta imagine are ca valoare width variabila $procent si %.....

Scriptul de votare folosit in tutorial il puteti descarca de la adresa:
http://www.oriceon.com/tutorial/descarca/exemple/ex4-votare.zip

Exercitiul 5

Realizarea unui sistem de cautare, a unor date, in baza de date

Problema:

Se cer urmatoarele campuri: titlu (tipul text), adresa (tipul text), si descriere (tipul textarea).

Cerinta:

Toate campurile sa fie obligatorii, sa nu contina mai putin de 2 caractere, sau mai mult de 255.

Rezolvare:

Prima si prima oara trebuie sa realizam baza de date.
Intram in phpMyAdmin si cream o baza de date cu numele cautare apoi tabela intrari cu 5 coloane si anume:

id | titlu | adresa | descriere | vizite

(Dupa cum am invatat mai sus in capitolul MySQL, campul ID trebuie sa fie de tipul INT, auto_increment si primary, apoi campurile: titlu VARCHAR(60), adresa VARCHAR(255), descriere VARCHAR(255), vizite BIGINT(10)).

Your SQL query has been executed successfully (Query took 0.1800 sec)

SQL query:

```
CREATE TABLE `intrari` (
  `id` INT( 11 ) NOT NULL AUTO_INCREMENT ,
  `titlu` VARCHAR( 255 ) NOT NULL DEFAULT '',
  `adresa` VARCHAR( 255 ) NOT NULL DEFAULT '',
  `descriere` VARCHAR( 255 ) NOT NULL DEFAULT '',
  `vizite` BIGINT( 10 ) NOT NULL DEFAULT '0',
  UNIQUE KEY `id` ( `id` )
)
```

[Edit] [Create PHP Code]

	Field	Type	Collation	Attributes	Null	Default	Extra	Action
	id	int(11)			No		auto_increment	
	titlu	varchar(255)	latin1_swedish_ci		No			
	adresa	varchar(255)	latin1_swedish_ci		No			
	descriere	varchar(255)	latin1_swedish_ci		No			
	vizite	bigint(10)			No	0		

Dupa ce ati realizat baza de date, intrati in directorul radacina al serverului dvs (daca folositi EasyPHP este folderul www) si creati un folder cu numele **cautare**.

Creati un fisier nou, in folderul cautare, numiti-l **config.php**, apoi introduceti codul de mai jos.

```php
<?php
session_start();
set_time_limit(0);
error_reporting(E_ALL);

// Informatii baza de date

    $AdresaBazaDate = "localhost";
    $UtilizatorBazaDate = "root";
    $ParolaBazaDate = "parola_baza";
    $NumeBazaDate = "cautare";

$conexiune = mysql_connect($AdresaBazaDate,$UtilizatorBazaDate,$ParolaBazaDate)
or die("Nu ma pot conecta la MySQL!");
mysql_select_db($NumeBazaDate,$conexiune) or die("Nu gasesc baza de date!");

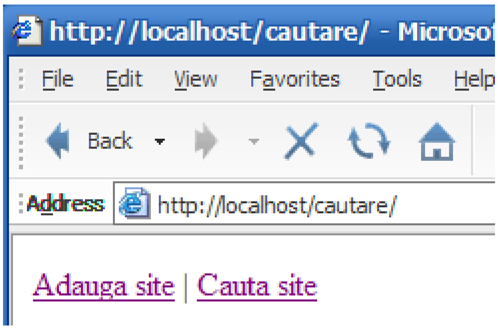
function addentities($data){
    if(trim($data) != ''){
    $data = htmlentities($data, ENT_QUOTES);
    return str_replace('\\', '&#92;', $data);
    } else return $data;
} // End addentities() --------------

?>
```

Acesta fiind fisierul de configurare.

In continuare, realizati un fisier **index.php**, apoi introduceti codul de mai jos si vizualizati in browser.

```php
<?php
echo '<a href="adauga.php">Adauga site</a> | <a href="cauta.php">Cauta site</a>';
?>
```



Dupa cum observati, avem 2 legaturi catre pagina **adauga.php** si cea de cautare **cauta.php**

Realizati fisierul **adauga.php**, introduceti codul de mai jos si testati in browser.

```php
<?php
require_once 'config.php';

if(!isset($_GET['pag'])) $_GET['pag'] = '';
if(!isset($_SESSION['titlu'])) $_SESSION['titlu'] = '';
if(!isset($_SESSION['adresa'])) $_SESSION['adresa'] = '';
if(!isset($_SESSION['descriere'])) $_SESSION['descriere'] = '';

switch($_GET['pag']) {

case '':
echo '<form name="adauga" action="adauga.php?pag=verifica" method="post">
      Titlu: <br> <input type="text" name="titlu" value="'.$_SESSION['titlu'].'"><br><br>
```

```
      Adresa: <br> <input type="text" name="adresa"
value="'.$_SESSION['adresa'].'"><br><br>
      Descriere: <br> <textarea name="descriere" rows="6" cols="45"
value="'.$_SESSION['descriere'].'">'.$_SESSION['descriere'].'</textarea><br><br>
      <input type="submit" name="Adauga" value="Adauga">
     </form>';
break;

case 'verifica':
$_SESSION['titlu'] = $_POST['titlu'];
$_SESSION['adresa'] = $_POST['adresa'];
$_SESSION['descriere'] = $_POST['descriere'];

if(($_SESSION['titlu'] == '') || (strlen($_SESSION['titlu']) < 2) ||
(strlen($_SESSION['titlu']) > 255) || ($_SESSION['adresa'] == '') ||
(strlen($_SESSION['adresa']) < 2) || (strlen($_SESSION['adresa']) > 255) ||
($_SESSION['descriere'] == '') || (strlen($_SESSION['descriere']) < 2) ||
(strlen($_SESSION['descriere']) > 255)) {

echo 'Completeaza corect campurile. <br>
      Vezi daca: ai completat campurile, daca ai scris mai mult de 2 caractere si mai
putin de 255<br><br>
    Apasa <a href="adauga.php">aici</a> pentru a te intoarce.';
} else {

$cerereSQL = "INSERT INTO `intrari` (`titlu`, `adresa`, `descriere`)
              VALUES ('".addentities($_SESSION['titlu'])."',
'".addentities($_SESSION['adresa'])."', '".addentities($_SESSION['descriere'])."');";
mysql_query($cerereSQL);

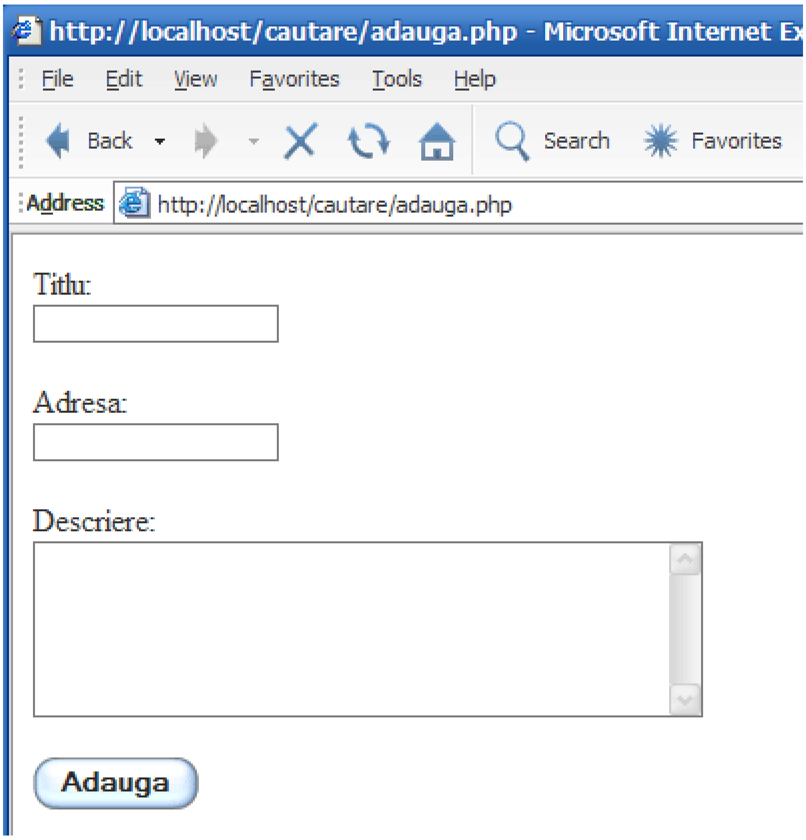
$_SESSION['titlu'] = '';
$_SESSION['adresa'] = '';
$_SESSION['descriere'] = '';

echo 'Am introdus datele in baza de date. <br>
      Apasa <a href="index.php">aici</a> pentru a te intoarce la pagina principala.';
}

break;

}

?>
```



Observati formularul prin care vom adauga datele in baza de date.

Aruncand o privire asupra codului, vedem ca ne-am folosit de instructionea switch si case pentru a delimita paginile, if si else pentru a conditiona rezultatele.

In case-ul default, `case '':`, avem formularul cu campurile: titlu, adresa, descriere iar ca valoare a acestora, avem setate sesiunile corespunzatoare lor (amintindu-va de explicatiile exercitiilor de mai sus, setam o sesiune pentru a pastra datele introduse de utilizator).

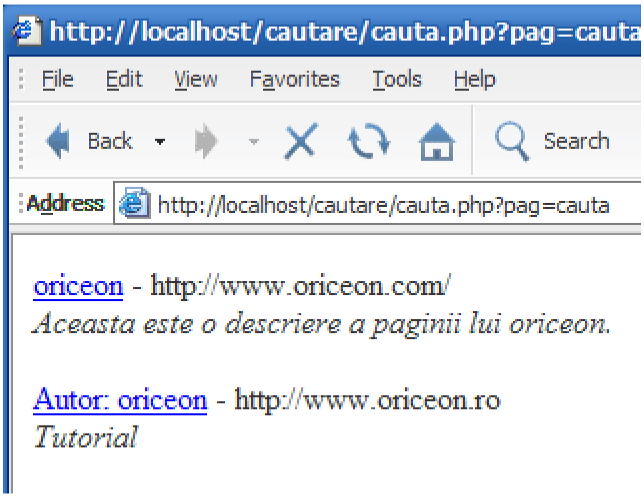
In case-ul `'verifica'`, observati setarea sesiunilor cu datele trimise de utilizator prin metoda $_POST, instructiunea if, in care avem conditiile, apoi instructiunea else ce va executa introducerea in baza de date, golirea sesiunilor folosite si afisarea unui text in pagina.

Ca protectie a introducerii datelor, am folosit functia addentities(); (vezi capitolul securitate).

Introduceti cateva adrese in baza de date, pentru a putea testa mai incolo cautarea.

Dupa ce ati introdus cateva adrese, copiati pagina **cautare.php** din arhiva **ex5-cautare.zip** descarcata de pe http://www.oriceon.com/tutorial/descarca/exemple/ex5-cautare.zip sau direct din arhiva descarcata o data cu tutorialul.

Apasati pe legatura “Cauta site” si introduceti un cuvant pentru cautare.



Observati rezultatele; cautarea s-a efectuat in baza de date, doar in celula titlu.

Deschideti fisierul **cautare.php** in editorul dumneavoastra preferat, si observati, ca noutate, comanda SQL:

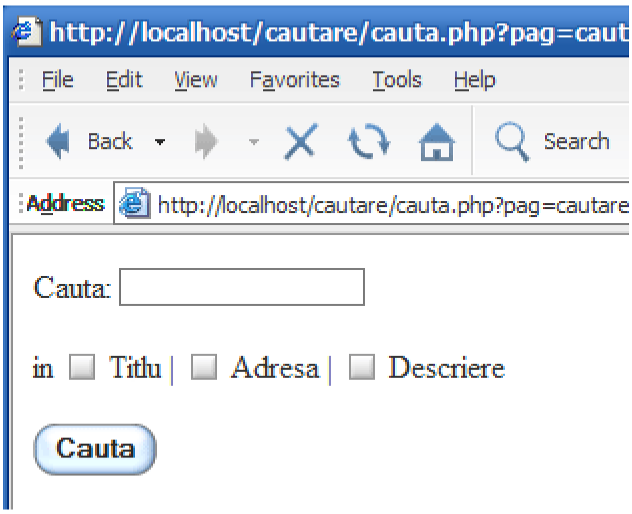
```
$cerereSQL = 'SELECT * FROM `intrari` WHERE `titlu` LIKE
"%'.addentities($_POST['cauta']).'%"';
```

Aceasta comanda SQL, selecteaza tot din baza de date unde campul titlu contine cuvantul trimis prin formular $_POST['cauta'].

De exemplu, daca ati introdus ca titlu, oriceonblablabla, si in formularul de cautare ati scris doar cuvantul oriceon, cautarea va rezulta si oriceonblablabla deoarece cuvantul oriceon a fost gasit in alcatuirea celui oriceonblablabla.

Aceasta este o simpla cautare in baza de date, dupa titlu, insa puteti realiza si o cautare mai avansata dupa titlu, adresa sau descriere.

Analizand case-ul `'cautare-avansata':`, observam scriptul pentru cautare avansata.



Am jonglat putin cu constructiile if si else, in functie de ce optiune era bifata:

```
if(($_POST['in1'] != 'titlu') && ($_POST['in2'] != 'adresa') && ($_POST['in3'] !=
'descriere')) {
$cerereSQL = 'SELECT * FROM `intrari` WHERE `titlu` LIKE
"%'.addentities($_POST['cauta']).'%"';
```

```
$in = '';
}
```

Aceasta parte de cod se interpreteaza cam asa:

Daca valoarea post in1 nu este egala cu titlu, si valoarea post in2 nu este egala cu adresa, si valoarea post 3 nu este egala cu descriere {

creaza o variabila cerereSQL care sa caute doar in titlu dupa textul dat,
creaza o variabila cu numele “in” cu valoare nula

}

Cu alte cuvinte, daca nu selectati nici o optiune, cautarea se va efectua ca cea anterioara, adica doar in campul titlu.

```
elseif(($_POST['in1'] == 'titlu') && ($_POST['in2'] != 'adresa') && ($_POST['in3'] !=
'descriere')) {
$cerereSQL = 'SELECT * FROM `intrari` WHERE `titlu` LIKE
"%'.addentities($_POST['cauta']).'%"';
$in = 'titlu';
}
```

Daca valoarea post in1 este egala cu titlu (adica daca optiunea titlu a fost selectata), si daca valoarile post-ului 2 si 3 nu sunt selectate {

creaza o variabila cerereSQL care sa caute in titlu dupa textul dau
Creaza o variabila cu numele “in” cu valoarea titlu

}

....
....

```
elseif(($_POST['in1'] == 'titlu') && ($_POST['in2'] == 'adresa') && ($_POST['in3'] !=
'descriere')) {
$cerereSQL = 'SELECT * FROM `intrari` WHERE `titlu` LIKE
"%'.addentities($_POST['cauta']).'%" AND `adresa` LIKE
"%'.addentities($_POST['cauta']).'%"';
$in = 'titlu, adresa';
}
```

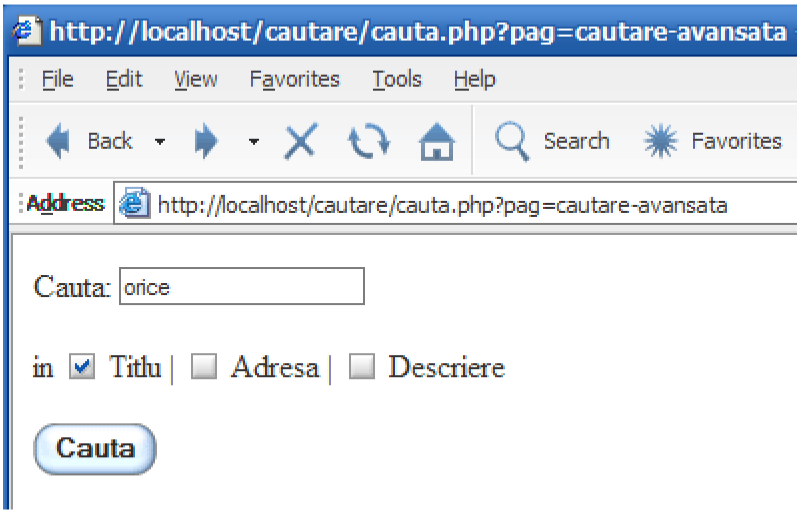
Daca valoarea post in1 este egala cu titlu (adica daca optiunea titlu a fost selectata), si daca valoarea post-ului in2 este egala cu adresa (adica daca si optiunea adresa a fost bifata), si daca valoarea post-ului in 3 nu este descriere {

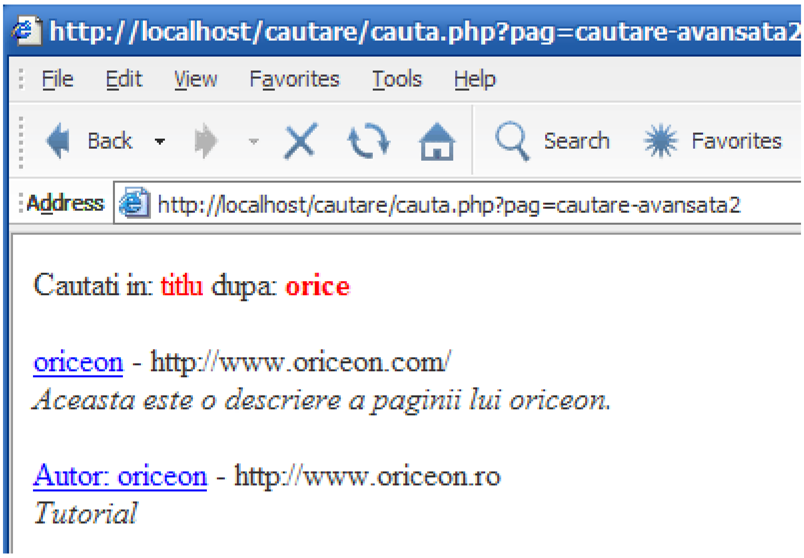
creaza o variabila cerereSQL care sa caute in titlu si adresa dupa textul dau
Creaza o variabila cu numele “in” cu valoarea titlu, adresa

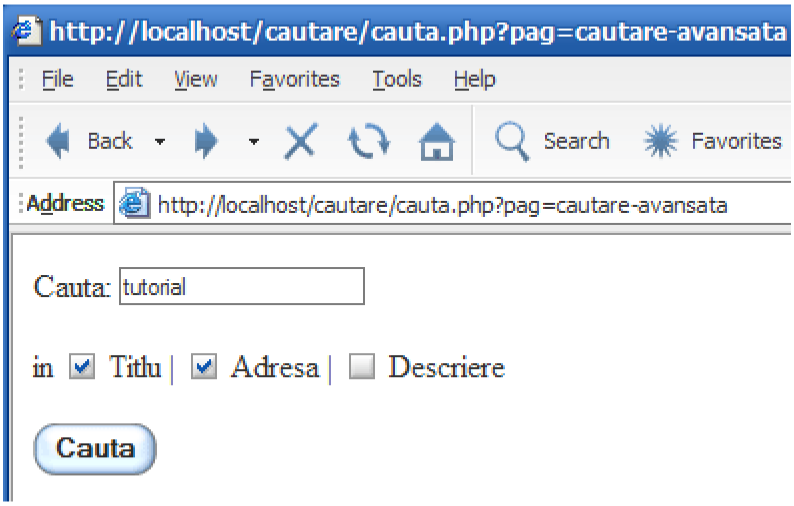
}

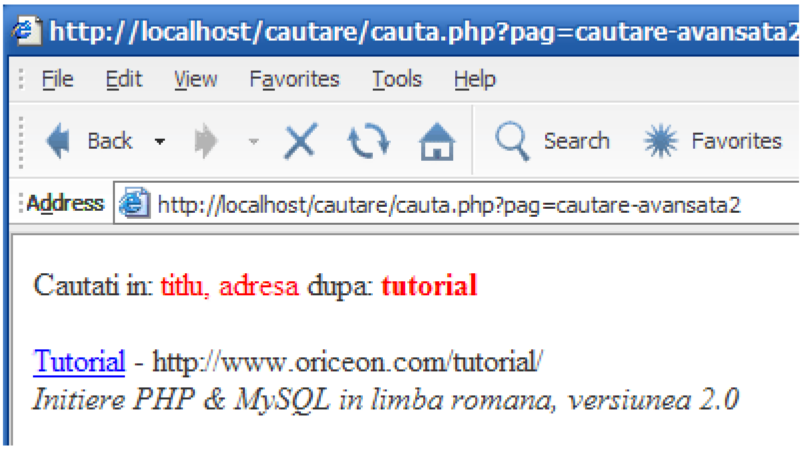
....
....

Efectuati un test in browser, in functie de adresele introduse de dumneavoastra in formular, bifati optiunile pe rand si testati.

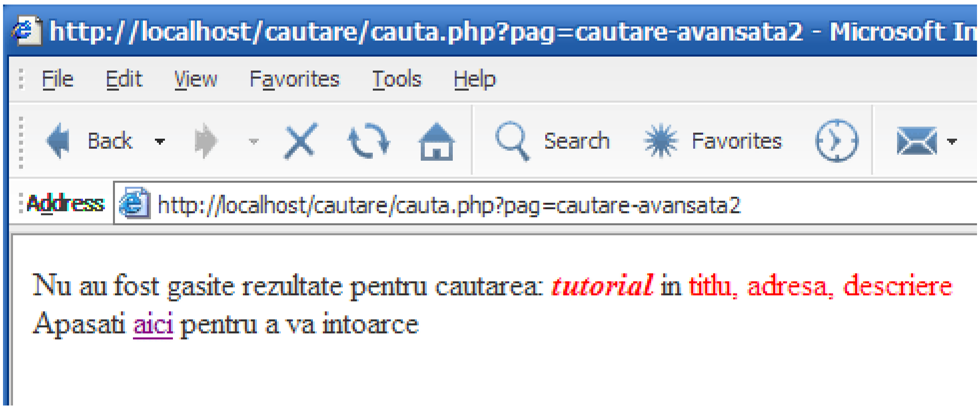

http://localhost/cautare/cauta.php?pag=cautare-avansata
File Edit View Favorites Tools Help
Back Search Favorites
Address http://localhost/cautare/cauta.php?pag=cautare-avansata
Cauta: orice
in Titlu | Adresa | Descriere
Cauta


http://localhost/cautare/cauta.php?pag=cautare-avansata2
File Edit View Favorites Tools Help
Back Search Favorites
Address http://localhost/cautare/cauta.php?pag=cautare-avansata2
Cautati in: titlu dupa: orice
oriceon - http://www.oriceon.com/
Aceasta este o descriere a paginii lui oriceon.
Autor: oriceon - http://www.oriceon.ro
Tutorial


http://localhost/cautare/cauta.php?pag=cautare-avansata
File Edit View Favorites Tools Help
Back Search Favorites
Address http://localhost/cautare/cauta.php?pag=cautare-avansata
Cauta: tutorial
in Titlu | Adresa | Descriere
Cauta



Sa luam un exemplu in care nu gaseste rezultate in baza de date,



Va las pe voi sa descoperiti de ce am setat variabila in, si pentru ce am folosit-o.

Scriptul de cautare folosit in tutorial il puteti descarca de la adresa:
http://www.oriceon.com/tutorial/descarca/exemple/ex5-cautare.zip

Exercitiul 6

Realizarea unui sistem de blocare acces al unui utilizator asupra site-ului

Problema:

Se cer 2 pagini, index.php si admin.php. In pagina admin.php trebuie sa aveti urmatoarele campuri: utilizator sau adresa IP (tipul text), si motiv (tipul text).

Cerinta:

Toate campurile sa fie obligatorii, sa nu contina mai putin de 2 caractere, sau mai mult de 255.

Rezolvare:

Cum ne-am invatat pana acum, prima data trebuie sa cream baza de date, insa acum vom lucra pe aceeasi baza de date cu care am lucrat si pentru exemplul cu autentificarea.
Intram in phpMyAdmin, selectam baza de date cu numele autentificare, apoi creati o noua tabela cu numele acces_interzis cu 3 coloane si anume:

id | interzis | motiv

(Dupa cum am invatat mai sus in capitolul MySQL, campul ID trebuie sa fie de tipul INT, auto_increment si primary, apoi campurile: interzis CHAR(60), motiv CHAR(255)).

Create new table on database autentificare:

Name: acces_interzis

Fields: 3

Go

Table acces_interzis has been created.

SQL query:

```
CREATE TABLE `acces_interzis` (
  `id` INT NOT NULL AUTO_INCREMENT ,
  `interzis` VARCHAR( 60 ) NOT NULL ,
  `motiv` VARCHAR( 255 ) NOT NULL ,
  PRIMARY KEY ( `id` )
) TYPE = MYISAM ;
```

[Edit] [Create PHP Code]

	Field	Type	Collation	Attributes	Null	Default	Extra	Action
☐	id	int(11)			No		auto_increment	
☐	interzis	varchar(60)	latin1_swedish_ci		No			
☐	motiv	varchar(255)	latin1_swedish_ci		No			

Dupa ce ati realizat baza de date, intrati in directorul radacina al serverului dvs (daca folositi EasyPHP este folderul www) si creati un folder cu numele **interzicere**.

Creati un fisier nou in folderul interzicere, numiti-l **config.php**, apoi introduceti codul de mai jos.

```php
<?php
session_start();
set_time_limit(0);
error_reporting(E_ALL);

// Informatii baza de date

    $AdresaBazaDate = "localhost";
    $UtilizatorBazaDate = "root";
    $ParolaBazaDate = "parola_baza";
    $NumeBazaDate = "autentificare";

$conexiune = mysql_connect($AdresaBazaDate,$UtilizatorBazaDate,$ParolaBazaDate)
or die("Nu ma pot conecta la MySQL!");
mysql_select_db($NumeBazaDate,$conexiune) or die("Nu gasesc baza de date!");

function addentities($data){
    if(trim($data) != ''){
    $data = htmlentities($data, ENT_QUOTES);
    return str_replace('\\', '&#92;', $data);
    } else return $data;
} // End addentities() --------------

?>
```

Dupa ce ati creat fisierul de configurare, realizati un alt fisier cu numele **index.php**, intorduceti codul de mai jos si testati in browser.

```php
<?php
require_once 'config.php';

$_SESSION['utilizator'] = 'oriceon';
$adresa_ip = $_SERVER['REMOTE_ADDR'];

$cerereSQL = 'SELECT * FROM `acces_interzis` WHERE interzis="'.$_SESSION['utilizator'].'"
OR interzis="'.$adresa_ip.'"';
$rezultat = mysql_query($cerereSQL);
if(mysql_num_rows($rezultat) > 0) {
    while($rand = mysql_fetch_array($rezultat)) {
    $motiv = $rand['motiv'];
    }
}

if(isset($motiv)) {
echo 'Acces interzis asupra paginii, motivul: <i>'.$motiv.'</i>';
} else {
echo 'Continutul paginii.<br>
      Apasa <a href="admin.php">aici</a> pentru a interzice acces-ul asupra paginii.';
}

?>
```

Observati faptul ca la inceputul paginii avem setata sesiunea cu utilizatorul logat (acum am definit-o manual, normal aceasta sesiune o setati atunci cand utilizatorul se autentifica, pagina index.php fiind una din paginile protejate, prin urmare vom avea sesiunea cu numele utilizatorului).

Adresa IP a utilizatorului o luam cu ajutorul variabilei globale $_SERVER, REMOTE_ADDR.

Dupa ce am obtinut utilizatorul sau adresa ip a acestuia, avem comanda SQL care selecteaza tot din tabela acces_interzis unde interzis este utilizator ori interzis este adresa ip.

Daca returneaza rezultate, setam variabila $motiv, cu valoarea randului motiv din baza de date, apoi realizam o constructie if cu conditia: daca este setata variabila motiv, inseamna ca au fost gasite rezultate, inseamna ca utilizatorul apare in baza de date ca fiind blocat, prin urmare afisam un mesaj de eroare.
Daca nu, afisam continutul paginii web.

Creatii o pagina cu numele **admin.php**, introduceti codul de mai jos, apoi testati in browser.

```
<?php
require_once 'config.php';

if(!isset($_GET['pag'])) $_GET['pag'] = '';
switch($_GET['pag']) {

case '':
echo '<form name="adauga" action="admin.php?pag=verifica" method="post">
      Utilizator sau Adresa IP <br> <input type="text" name="interzis"><br><br>
      Motiv <br> <input type="text" name="motiv"><br><br>
      <input type="submit" name="Adauga" value="Adauga">
      </form>';
break;

case 'verifica':

if(($_POST['interzis'] == '') || (strlen($_POST['interzis']) < 2) ||
(strlen($_POST['interzis']) > 255) || ($_POST['motiv'] == '') || (strlen($_POST['motiv'])
< 2) || (strlen($_POST['motiv']) > 255)) {
echo 'Completeaza corect campurile. <br>
      Vezi daca: ai completat campurile, daca ai scris mai mult de 2 caractere si mai
putin de 255<br><br>
Apasa <a href="admin.php">aici</a> pentru a te intoarce.';
} else {

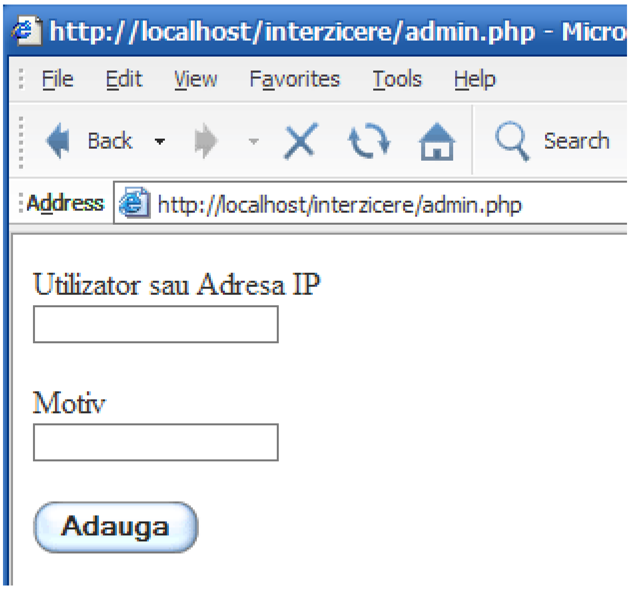
$cerereSQL = "INSERT INTO `acces_interzis` (`interzis`, `motiv`)
              VALUES ('".addentities($_POST['interzis'])."',
'".addentities($_POST['motiv'])."');";
mysql_query($cerereSQL);

echo 'Am introdus datele in baza de date. <br>
      Apasa <a href="index.php">aici</a> pentru a te intoarce la pagina principala.';
}

break;

}

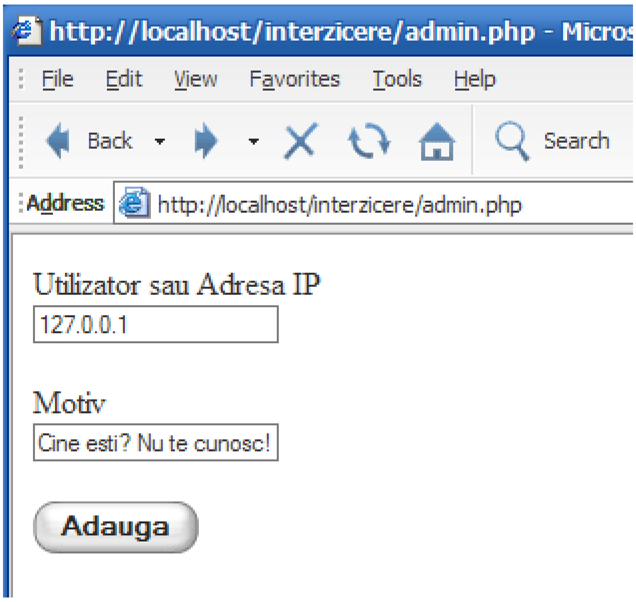
?>
```

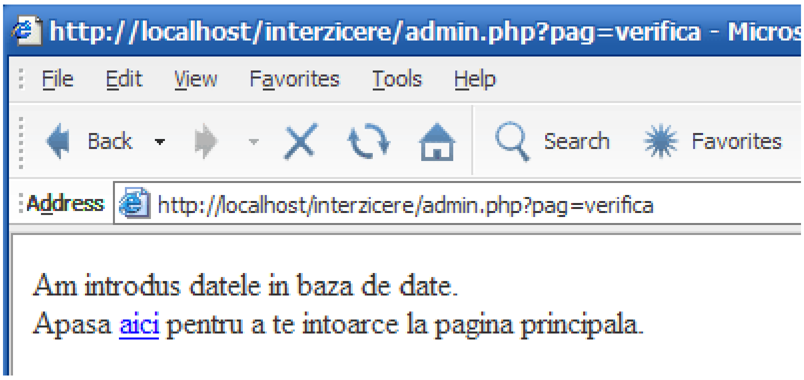


Observati faptul ca in case default, **case ''**:, avem formularul de blocare utilizator sau IP si o casuta cu motivul care va apare atunci cand un utilizator blocat acceseaza adresa.

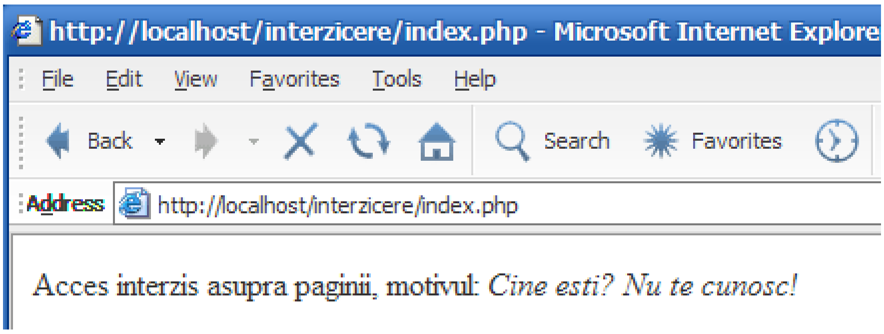
A II-a parte, cea cu validarea si inserarea in baza de date, va este cunoscuta.

Introduceti, spre exemplu, adresa dvs locala: 127.0.0.1 si un motiv apoi testati in browser accesand pagina principala index.php.





Accesati index.php si observati mesajul de eroare.



Scriptul de interzicere folosit in tutorial il puteti descarca de la adresa:
http://www.oriceon.com/tutorial/descarca/exemple/ex6-interzicere.zip

Exercitiul 7

Realizarea unui sistem de contorizare click-uri efectuate pe un link

Problema:

Se cere o pagina web in care sa se contorizeze click-urile efectuate pe link-urile din pagina.

Rezolvare:

Cum ne-am invatat pana acum, prima data trebuie sa cream baza de date, insa o avem deja creata in exercitiul cu scripturile de cautare.

Ne vom folosi de celula vizite din baza de date cautare, tabela intrari.

Intrati in directorul radacina al serverului dvs (daca folositi EasyPHP este folderul www) si creati un folder cu numele **contorizare**.

Copiati apoi fisierul **config.php** din folderul cautare in folderul contorizare.

Realizati apoi o pagina cu numele **index.php**, apoi introduceti codul de mai jos si testati in browser.

```
<?php
require_once('config.php');

if(!isset($_GET['pag'])) $_GET['pag'] = '';
switch($_GET['pag']) {

case '':
$cerereSQL = 'SELECT * FROM `intrari`';
$rezultat = mysql_query($cerereSQL);
while($rand = mysql_fetch_array($rezultat)) {
echo '<a href="index.php?pag=redirectionare&id='.$rand['id'].'">'.$rand['titlu'].'</a> -
'.$rand['adresa'].' <font color="blue" size="2">(vizualizari: '.$rand['vizite'].')</font>
<br> <i>'.$rand['descriere'].'</i> <br><br>';
}
break;

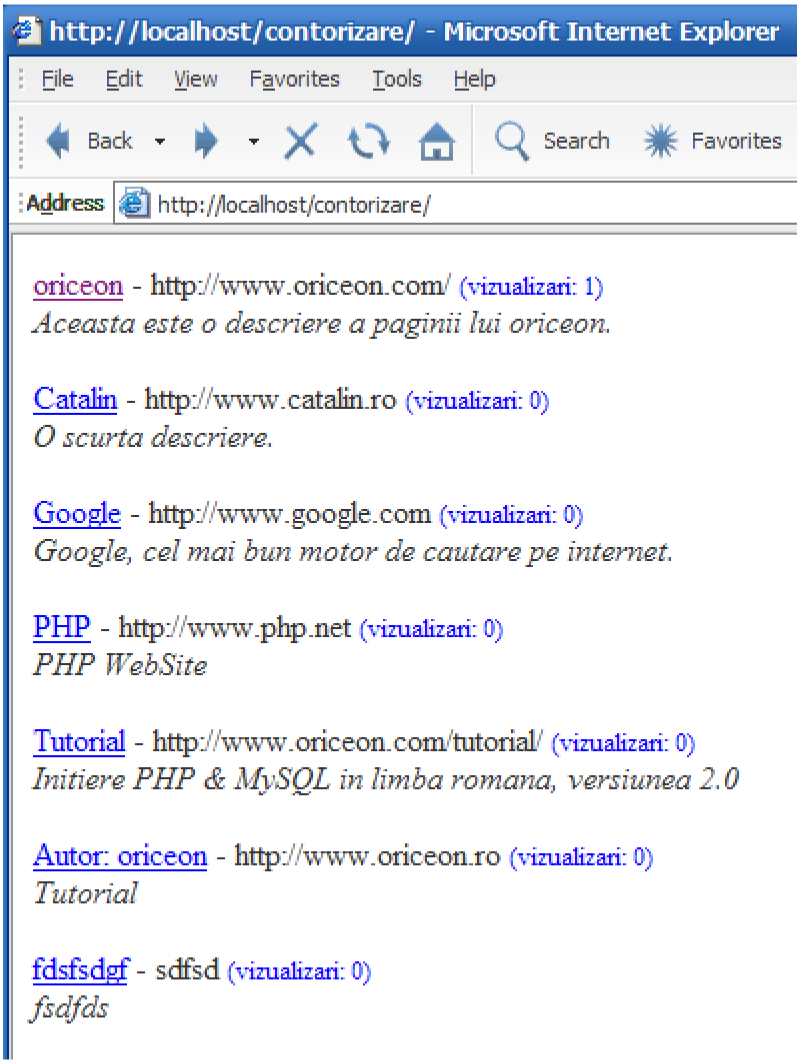
case 'redirectionare':

if(!is_numeric($_GET['id'])) {
echo 'ID nu este numeric, ce incerci sa faci?';
} else {
$cerereSQL = 'SELECT * FROM `intrari` WHERE id="'.$_GET['id'].'"';
$rezultat = mysql_query($cerereSQL);
while($rand = mysql_fetch_array($rezultat)) {
header("Location: ".$rand['adresa']."");
$cerereSQL = 'UPDATE `intrari` SET vizite="'.($rand['vizite']+1).'" WHERE
id="'.$rand['id'].'"';
$rezultat = mysql_query($cerereSQL);
}
}

break;

}

?>
```



Observati ca in primul case avem selectarea si afisarea in pagina a tuturor adreselor introduse din proiectul cautare; mai observati si transferarea prin:

```
<a href="index.php?pag=redirectionare&id='.$rand['id'].'">
```

a id-ului corespunsator fiecarei intrari.

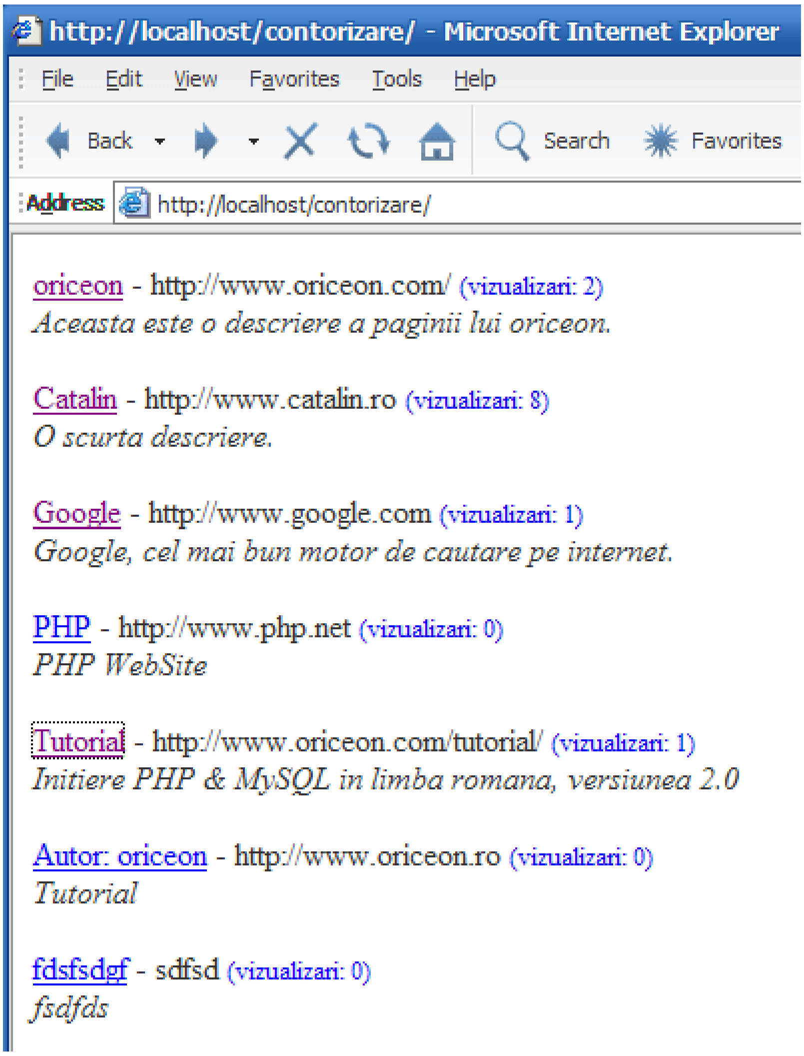
In al II-lea case avem verificarea daca valoarea trimisa prin $_GET este numerica. Daca nu este numerica, afisam un mesaj de eroare, iar daca este, selectam tot din baza de date unde id-ul este id-ul trimis prin get.

Redirectionam utilizatorul catre pagina selectata prin metota header Location.

```
header("Location: ".$rand['adresa']."");
```

Apoi updatam vizitele cu +1, unde id-ul este id-ul trimis prin $_GET.

Accesati din nou pagina principala index.php si observati rezultatele:


http://localhost/contorizare/ - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back
Search
Favorites
Address
http://localhost/contorizare/
oriceon - http://www.oriceon.com/ (vizualizari: 2)
Aceasta este o descriere a paginii lui oriceon.
Catalin - http://www.catalin.ro (vizualizari: 8)
O scurta descriere.
Google - http://www.google.com (vizualizari: 1)
Google, cel mai bun motor de cautare pe internet.
PHP - http://www.php.net (vizualizari: 0)
PHP WebSite
Tutorial - http://www.oriceon.com/tutorial/ (vizualizari: 1)
Initiere PHP & MySQL in limba romana, versiunea 2.0
Autor: oriceon - http://www.oriceon.ro (vizualizari: 0)
Tutorial
fdsfsdgf - sdfsd (vizualizari: 0)
fsdfds

Scriptul de contorizare folosit in tutorial il puteti descarca de la adresa:
http://www.oriceon.com/tutorial/descarca/exemple/ex7-contorizare.zip

Exercitiul 8

Realizarea unui sistem de cenzura cuvinte

Intrati in directorul radacina al serverului dvs (daca folositi EasyPHP este folderul www) si creati un folder cu numele **cenzura**.

Realizati apoi o pagina cu numele **index.php**, apoi introduceti codul de mai jos si testati in browser.

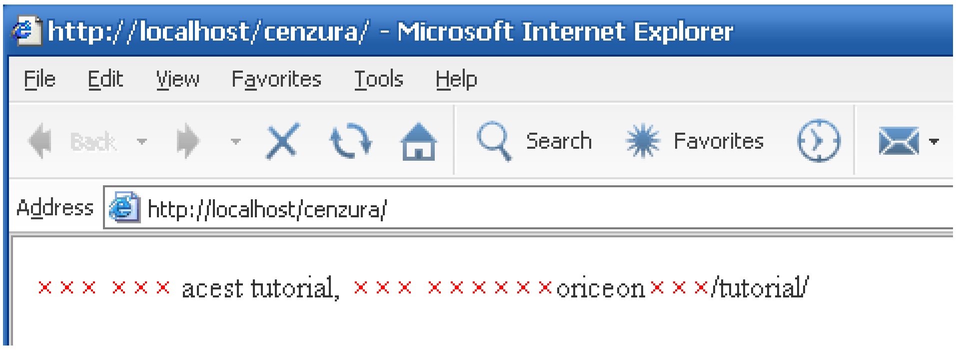
```php
<?php

$cuvinte_interzise = array("vand", "urgent", "vizitati", "http://", "www.", ".com");
$inlocuieste_cu = "<span style=\"color:red; font-style: italic;\">xxx</span>";

function cenzura($continut) {
    global $cuvinte_interzise, $inlocuieste_cu;
        foreach($cuvinte_interzise as $cuvinte) {
        $continut = eregi_replace($cuvinte, $inlocuieste_cu, $continut);
        }
return $continut;
}

echo cenzura("Vand urgent acest tutorial, vizitati http://www.oriceon.com/tutorial/");

?>
```



Observati cele 2 variabile care se afla in afara functiei.
Variabila `$cuvinte_interzise` contine cuvintele interzise (puteti adauga cate cuvinte vreti).
Variabila `$inlocuieste_cu` contine caracterele ce vor inlocui cuvintele gasite (puteti seta ce caracter vreti, o poza sau, orice).

In functia cenzura setam variabilele amintite mai sus, ca fiind globale (altfel nu vom putea folosi continutul celor doua variabile in functie) apoi ne folosim de **foreach** si expresia regulata **eregi_replace**.

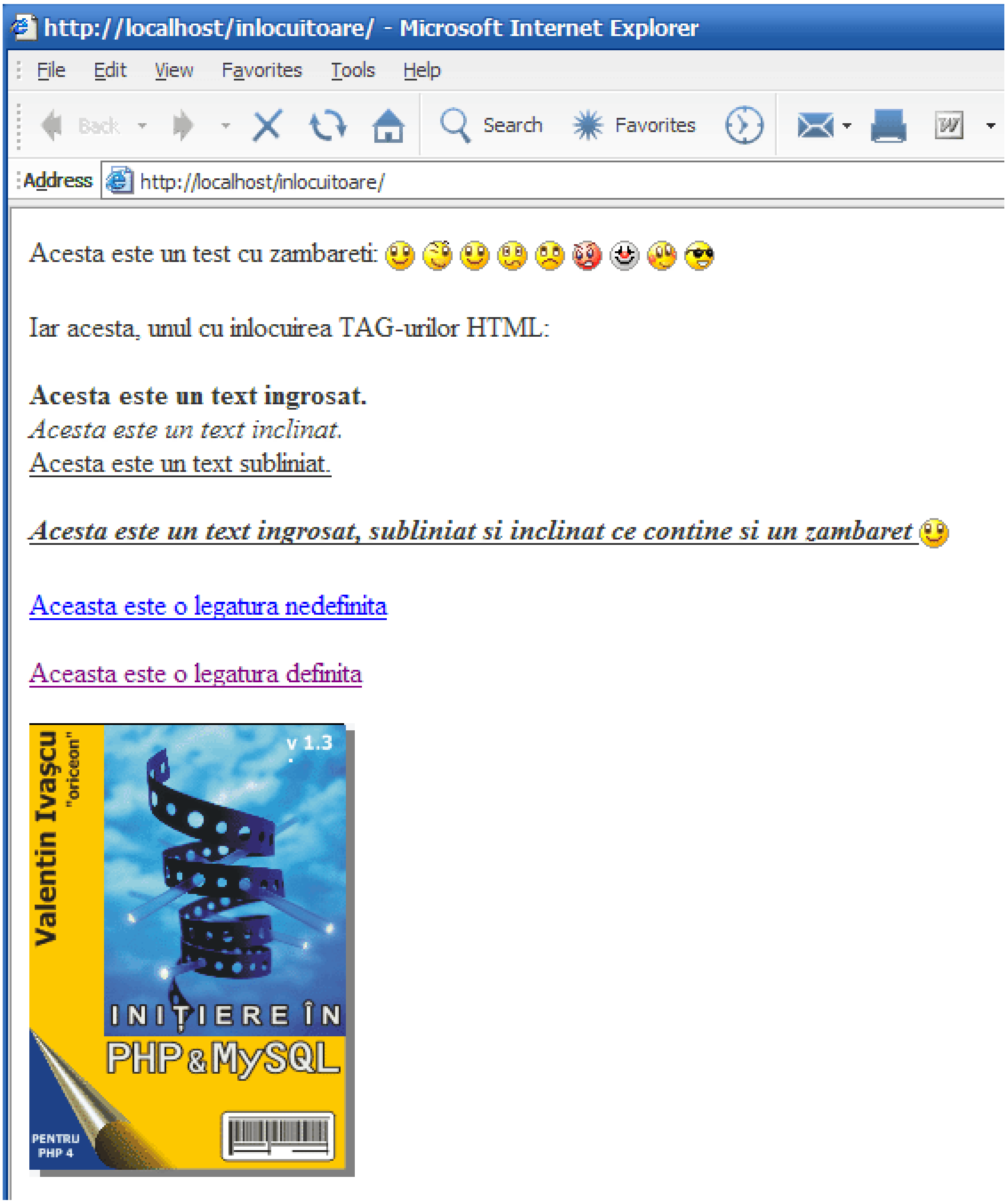
Scriptul de cenzura folosit in tutorial il puteti descarca de la adresa:
http://www.oriceon.com/tutorial/descarca/exemple/ex8-cenzura.zip

Exercitiul 9

Realizarea unui sistem de inlocuire coduri HTML si zambareti

Intrati in directorul radacina al serverului dvs (daca folositi EasyPHP este folderul www) si creati un folder cu numele **inlocuitoare**.

Copiati pagina **index.php** din arhiva **ex9-inlocuitoare.zip** descarcata de pe http://www.oriceon.com/tutorial/descarca/exemple/ex9-inlocuitoare.zip sau direct din arhiva descarcata o data cu tutorialul, analizati codul, apoi testati in browser.



Uitandu-va peste cod, observati functia cu numele formatare ce inlocuieste un string cu un altul.

In aceasta functie ne-am folosit de array-uri, de foreach, iar ca noutate observati functia eregi_replace ce ne ajuta sa inlocuim o valoare, cu o alta, dintr-un text dat (aceasta functie face parte din Expresiile Regulare insa momentan nu este cazul sa le dezbatem mai pe larg).

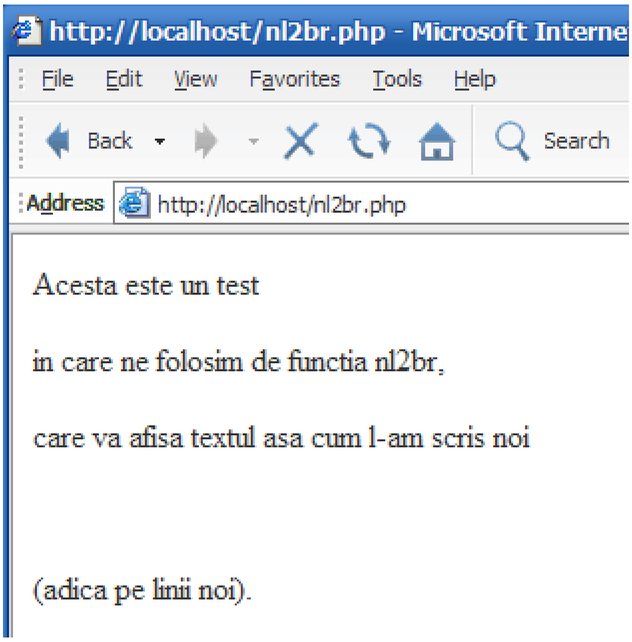
O alta noutate este folosirea functiei nl2br.
Aceasta functie returneaza un
 (un nou rand) la liniile noi date.

Creati o noua pagina cu numele **nl2br.php**, apoi introduceti codul de mai jos si testati in browser:

```
<?php
$text = 'Acesta este un test
in care ne folosim de functia nl2br,
care va afisa textul asa cum l-am scris noi


(adica pe linii noi).';
echo nl2br($text);

?>
```



Scriptul de inlocuitoare folosit in tutorial il puteti descarca de la adresa:
http://www.oriceon.com/tutorial/descarca/exemple/ex9-inlocuitoare.zip

Exercitiul 10

Realizarea unui sistem de paginare

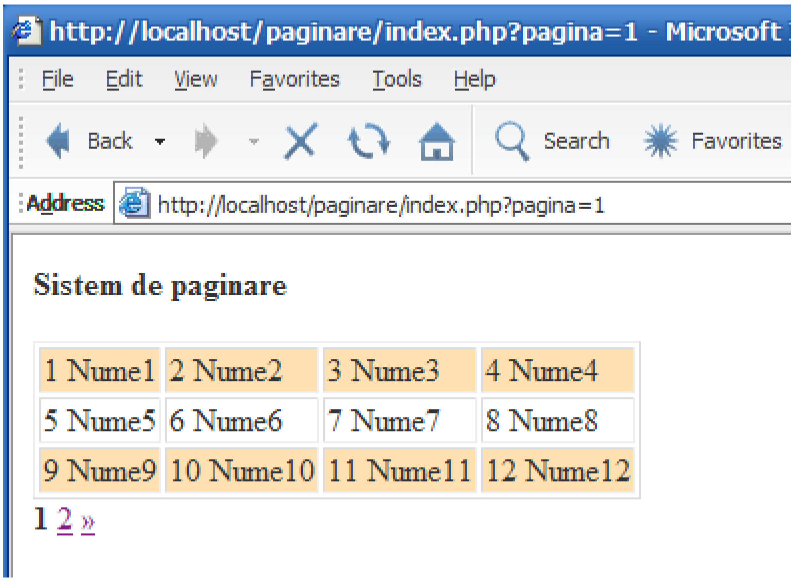
In acest exemplu veti invata cum sa realizati o paginare a rezultatelor extrase de dumneavoastra din baza de date.

Dupa ce ati descarcat scriptul de paginare de la adresa http://www.oriceon.com/tutorial/descarca/exemple/ex10-paginare.zip , dezarhivati si puneti folderul paginare in directorul radacina al serverului dumneavoastra.

Realizati o baza de date pentru el apoi modificati fisierul config.php si index.php cu datele necesare, apoi testati in browser si analizati scriptul din index.php.

Nu este nici o noutate, doar niste calcule matematice, niste folosiri ale variabilelor, ale constructiilor if si else.

Sunt foarte obosit ca sa il mai comentez, asa ca va las pe voi sa va puneti putin mintea la contributie, si daca ati fost atenti si ati parcurs tutorialul pana aici, veti putea intelege si acest exemplu.



Scriptul de paginare folosit in tutorial il puteti descarca de la adresa:
http://www.oriceon.com/tutorial/descarca/exemple/ex10-paginare.zip

Proiect 1

Acest proiect reprezinta o folosire a tuturor exemplelor date pana acum.

Va avea numele de “Sistem de noutati”, ce va cuprinde:

- un sistem de inregistrare utilizatori,
- sistem de votare,
- sistem de cautare simpla si avansata in stirile active,
- sistem de afisare stiri / cu paginare,
- posibilitatea administratorului si a editorilor de a arhiva/sterge/modifica o stire,
- protectie impotriva cuvintelor obscene, impotriva atacurilor,
- cod pentru a inlocui :) :P :(cu zambareti, cod pentru a inlocui unele tag-uri HTML,
- utilizatorul va putea trimite o stire, administratorul si editorul o va aproba,
- utilizatorul va castiga puncte in urma trimiterii stirilor,
- top 10 utilizatori (in functie de punctaje),
- administratorul poate interzice un utilizator, ip, poate interzice un cuvant a nu fi folosit la inregistrare de useri,
- afisarea ultimului utilizator inregistrat,
- afisarea celui mai lenes utilizator,
- in functie de punctaj, utilizatorul ajunge automat moderator insa numai cu aprobarea administratorului,
- statistici asurpa stirilor vizitate si asupra paginii.
- etc…

Proiectul va fi disponibil pe pagina http://www.oriceon.com/tutorial/descarca/proiect1.zip insa mometan lucrez la el.

In momentul in care il termin si il public, am sa anunt pe adresele forum-urilor date in prima pagina.

Parteneriat

Un SOC (**Si**stem de **O**rganizare a **C**ontinutului = **C**ontent **M**anagement **S**ystem) - realizat de un roman este usor, frumos si cuprinde foarte multe facilitati.
Il puteti descarca de la adresa http://cms.punctweb.com/

PHPEdit este cel mai frumos editor de PHP cu care am lucrat vreodata.
Vi-l recomant cu cea mai mare caldura si sa stiti ca puteti sa obtineti si o liceenta personala (chiar daca programul nu este gratuit), insa puteti sa o folositi doar in scop personal, nu si comercial.
Va invit sa vizitati pagina http://www.waterproof.fr/

MULTUMIRI

Astept parerile voastre si sugestii la email oriceon@yahoo.com

Versiunile ulterioare vor cuprinde mai multe exemple de scripturi PHP si explicatiile aferente.

Multumiri pentru: **i0nutzb, punctweb, Excallbvr, em@il, expertPHP, Paliu Catalin** pentru contributiile aduse asupra tutorialului.

Multumesc si lui birkoff pentru colaborare, acesta participand cu documentatia sa de pe http://www.php4.as.ro, in urma careia am imbunatatit unele capitole din acest tutorial.

Pentru parteneriat sau orice propunere, folositi aceeasi adresa de email ca cea de mai sus.

Pagina principala a acestui tutorial este: http://www.oriceon.com/tutorial/
Va astept aici pentru a urmari noile versiuni.

Cu stima,
Ivascu Valentin (oriceon)